Agenda

- **Work at University of Manitoba – Ravi Shrestha (15 min)**
- **Work at Rush University – Dale Stentz (15 min)**
- **Status and Update – Woon-Seng Choong (15 min)**
- **What’s New in the Latest Release v2.0 – Faisal Abu-Nimeh (20 min)**
- **Q & A**

Customized OpenPET firmware for a MR compatible PET insert

R. Shrestha[1,2] , M. S. Khan[3], C. J. Thompson[4], G. Stortz[5], G. Schellenberg[6], P. Kozlowski[7], F. Retiere[8], E. Shams[9], V. Sossi[5], J.D. Thiessen[10] and A. L. Goertzen[1,6]

[1]Department of Radiology, University of Manitoba, Winnipeg, Canada
[2]Cubresa Inc., Winnipeg, Canada
[3]Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, Canada
[4]Montreal Neurological Institute, McGill University, Montreal, Canada
[5]Department of Physics & Astronomy, University of British Columbia, Vancouver, Canada
[6]Department of Physics & Astronomy, University of Manitoba, Winnipeg, Canada
[7]Department of Radiology, University of British Columbia, Vancouver, Canada
[8]Detector Development Group, TRIUMF, Vancouver, Canada
[9]Biomedical Engineering Graduate Program, University of Manitoba, Winnipeg, Canada
[10]Imaging Program, Lawson Health Research Institute, London, Canada

Health Sciences Centre Winnipeg

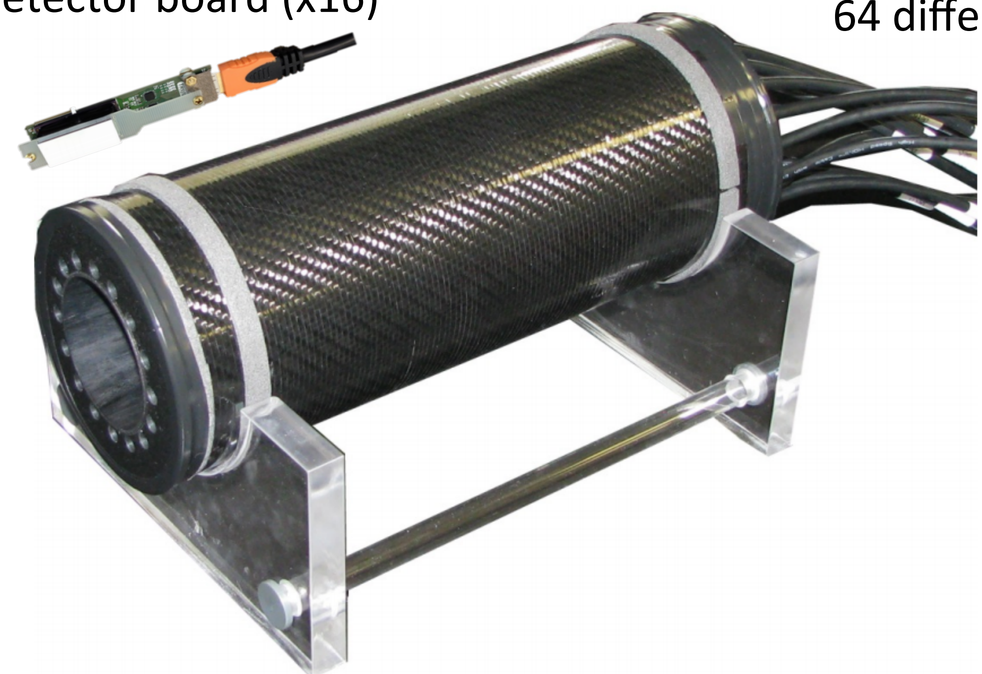
OpenPET Users Group Meeting, 2015

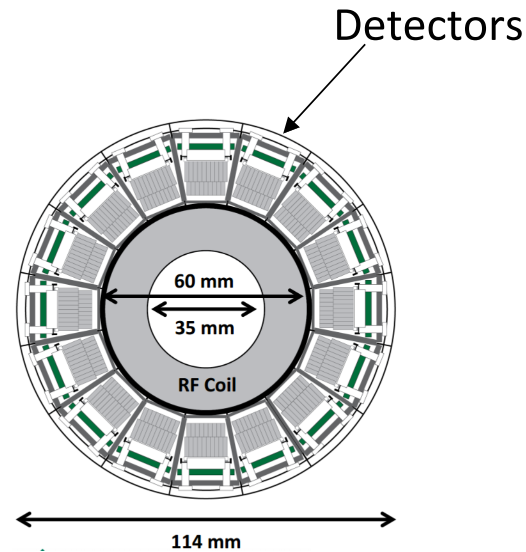
University of Manitoba

PET System Design

Detector board (x16)

16 HDMI cables carrying
64 differential analog signals





Cross section view of PET Insert

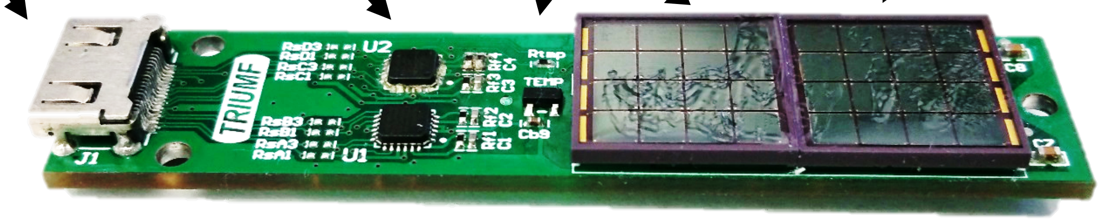
Detector design

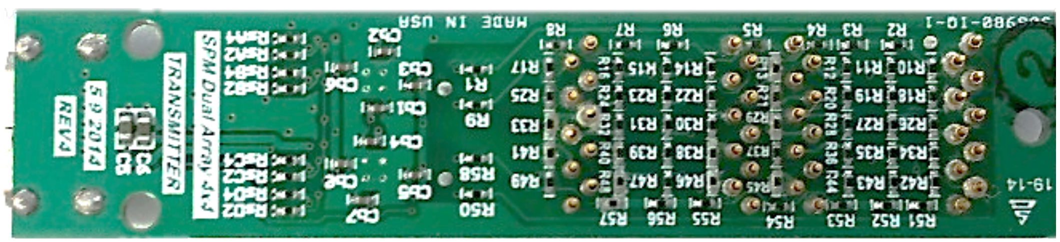
Mini HDMI connector

2x Differential amplifiers

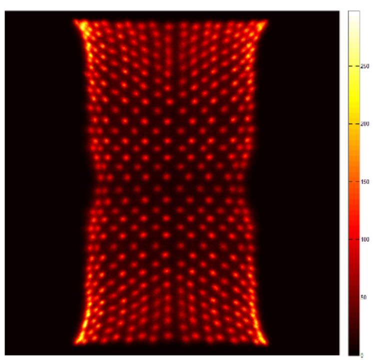
Temperature sensor

2x SensL SPMArray4B

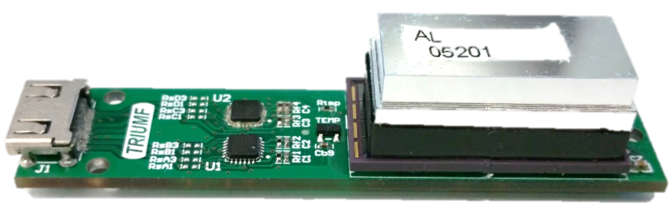




Resistor charge division network



Single detector flood image acquired using modified firmware



Detector with LYSO:Ce Dual-layer offset scintillator array installed

Limitations of initial OpenPET firmware

- Trigger on any analog channel results readout of all channels
  - This results significant dead time on front end
- 2124 bytes of data is generated per trigger. The packet structure is
  - 2 start words
  - 16 x 32 ADC sample words (supports up to 256 samples per trigger)
  - 16 TDC sample words
  - 1 end word
- Using this packet structure, we can get maximum ~18.8kcps @ 2.1kB/event over USB2.0 (40MB/s)
- No global reset for TDC counter
  - No time synchronization between detector boards
- 4 bit data bus is used for transferring events from detector board to support board
- Fast timing discriminator is not used

Customization of OpenPET firmware

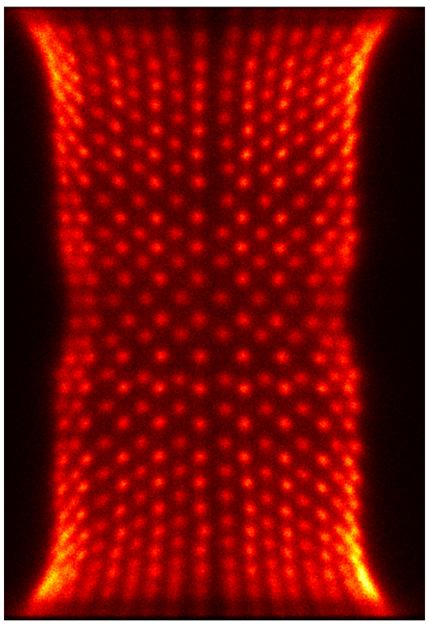
- Hard reset button is implemented to reset TDC counter globally across all detector boards
- Independent triggering of each detector block is implemented
- Both slow and fast discriminator is used to trigger event and capture time stamp respectively
- We implemented independent data bus for group of 4 ADC channels (one detector block)
  - This reduces dead time in the front end significantly
- We implemented customized singles mode with two different data packet format of 64 bits.
  - Calibration mode (ADC data only)
  - Operation mode (ADC+TDC data)
- Status packet is generated every second with detector event rate and system absolute time

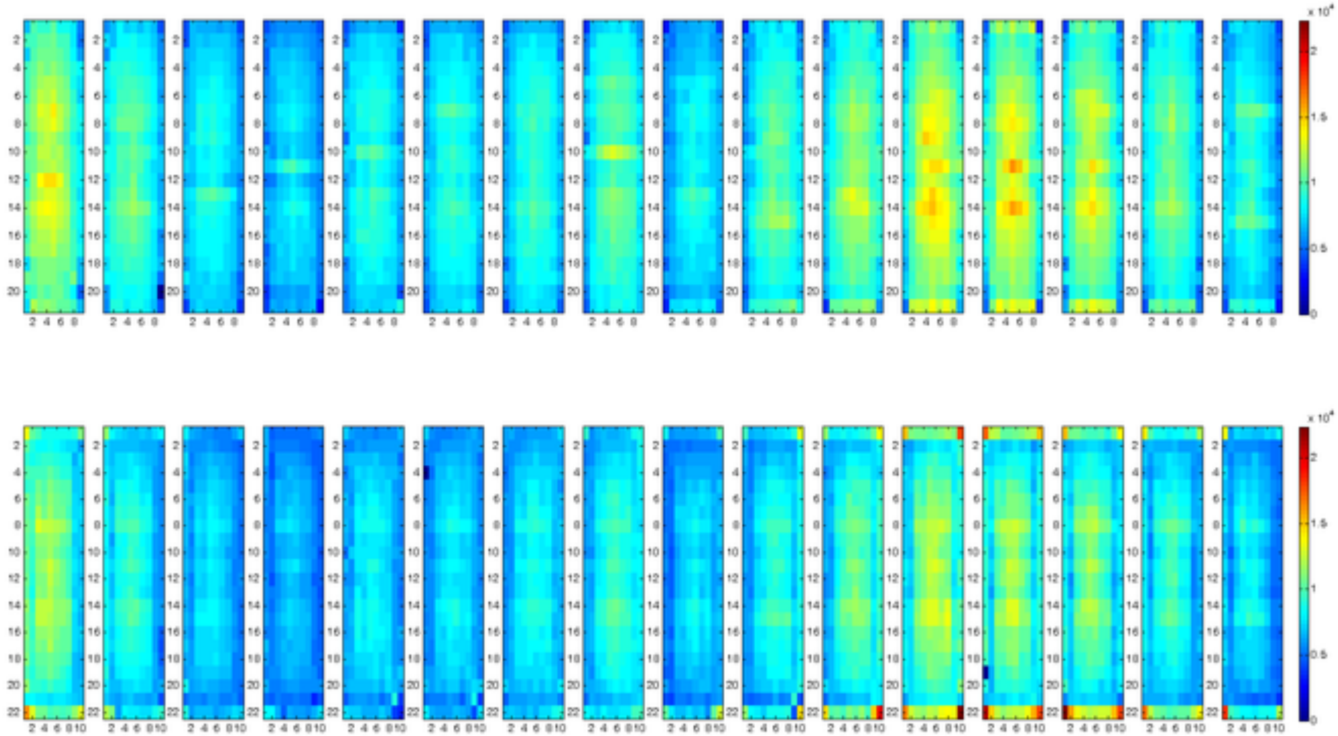
Summary of Firmware Modifications

		Initial Firmware	Modified Firmware
Detector Board (DB)	Operating Mode	Oscilloscope mode	Singles Mode with ADC only and ADC+TDC data
	Data packet size	2.1 kBytes per trigger (32 ADC samples per channel)	8 Bytes per trigger
	Discriminator	Slow discriminator	Slow discriminator for event trigger and fast discriminator for time stamp capture
	DB Time stamp synchronization	-	Timer synchronized across all DBs by using single external reset push button
	Trigger	Single trigger for all 16 ADC channels	Independent triggering per group of 4 ADC channels
	Data processing	-	Find ADC maximum value and calculate the (maximum-baseline) value
	Front End dead time	18µs	1.275µs
	Data path to IO FPGA	4 bits multiplexed with all ADC channels	16 bits, independent data path for each detector block
IO FPGA	Data path to main FPGA	4 bits (1 event transferred / 53.1µs)	16 bits (2 events transferred per 100ns)
	Event FIFO	-	Independent FIFO for each detector block and multiplexed event FIFO
Main FPGA	FIFO	-	FIFO optimized for handling 2 events at a clock
	Max. System Singles rate	18 kcps	5 Mcps

Detailed on firmware modification will be in the poster **M5DP-36 MIC Poster IV (Initial Results and Experience with an OpenPET Data Acquisition Platform with SiPM Based Detectors for a PET/MR System)**

Detector Flood Images and Crystal Efficiency Maps

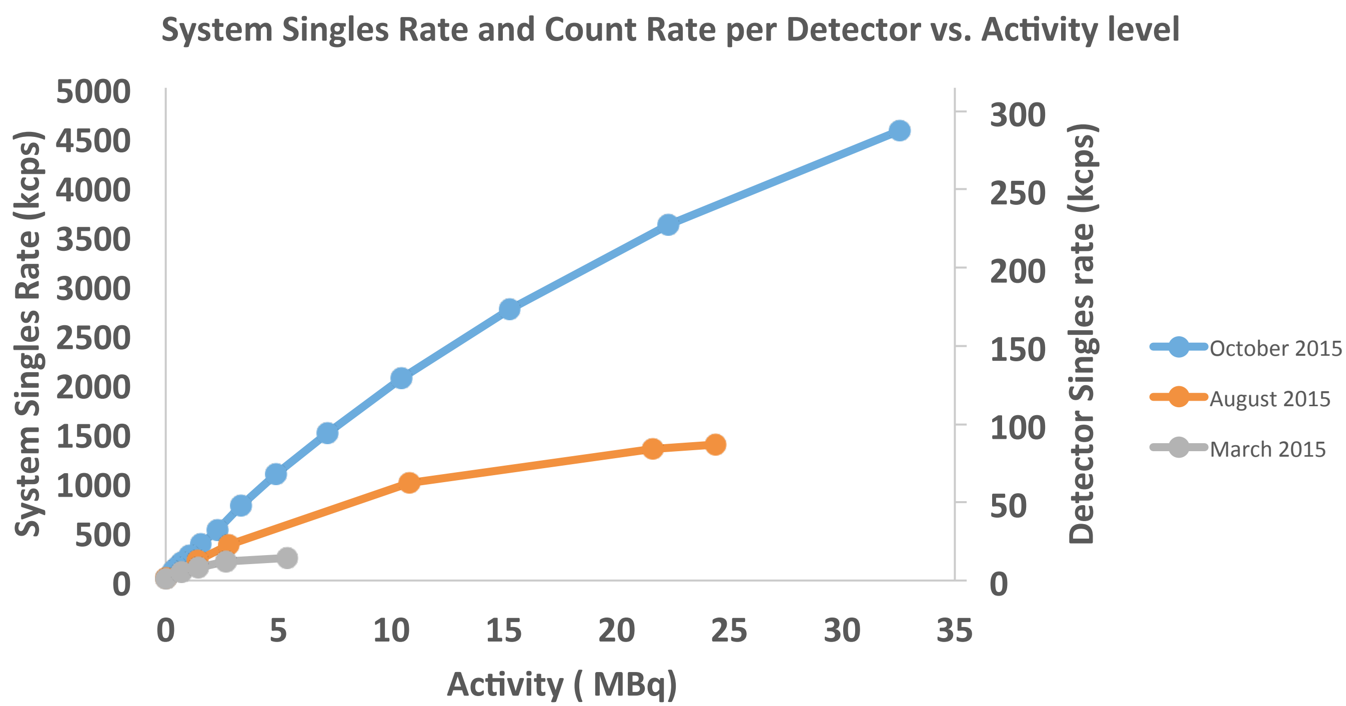




Detector Flood Image
from single detector

7.4 MBq (53.7 kcps per detector measured singles rate)

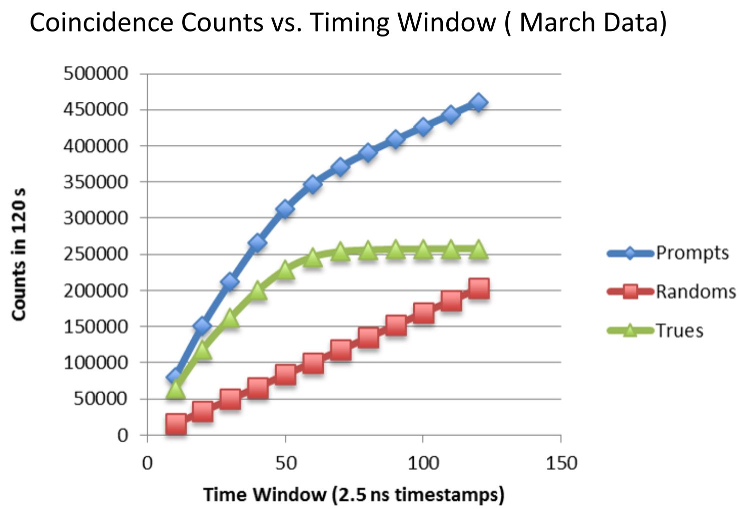
Results: System Singles Rate

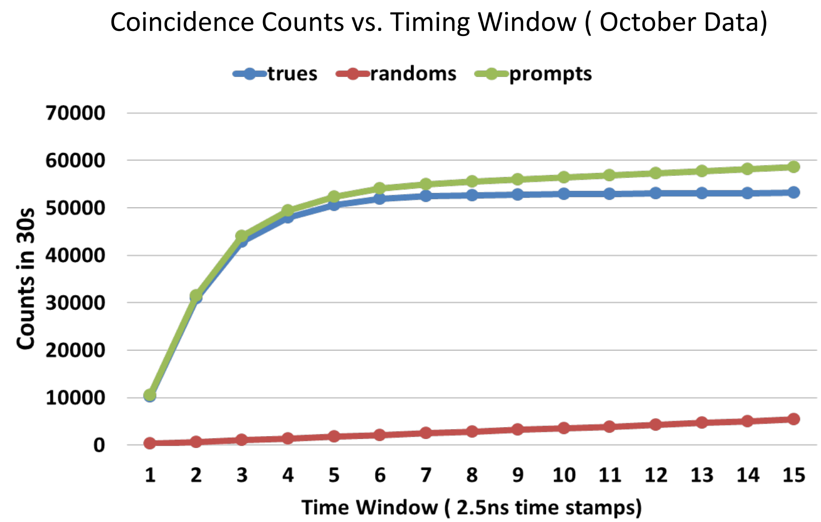


Total system singles rate plotted vs. activity in NEMA Nu4 mouse count rate phantom.

Firmware changes	Upgrades
March 2015	140 bytes data format , 4 bit data path, dead time: 18µS
August 2015	8 bytes data format, dead time: 2 to 4.4µS
October 2015	8 bytes data format with independent triggering, 16 bit parallel data path implemented, dead time: 1.275µS

Results: Timing window

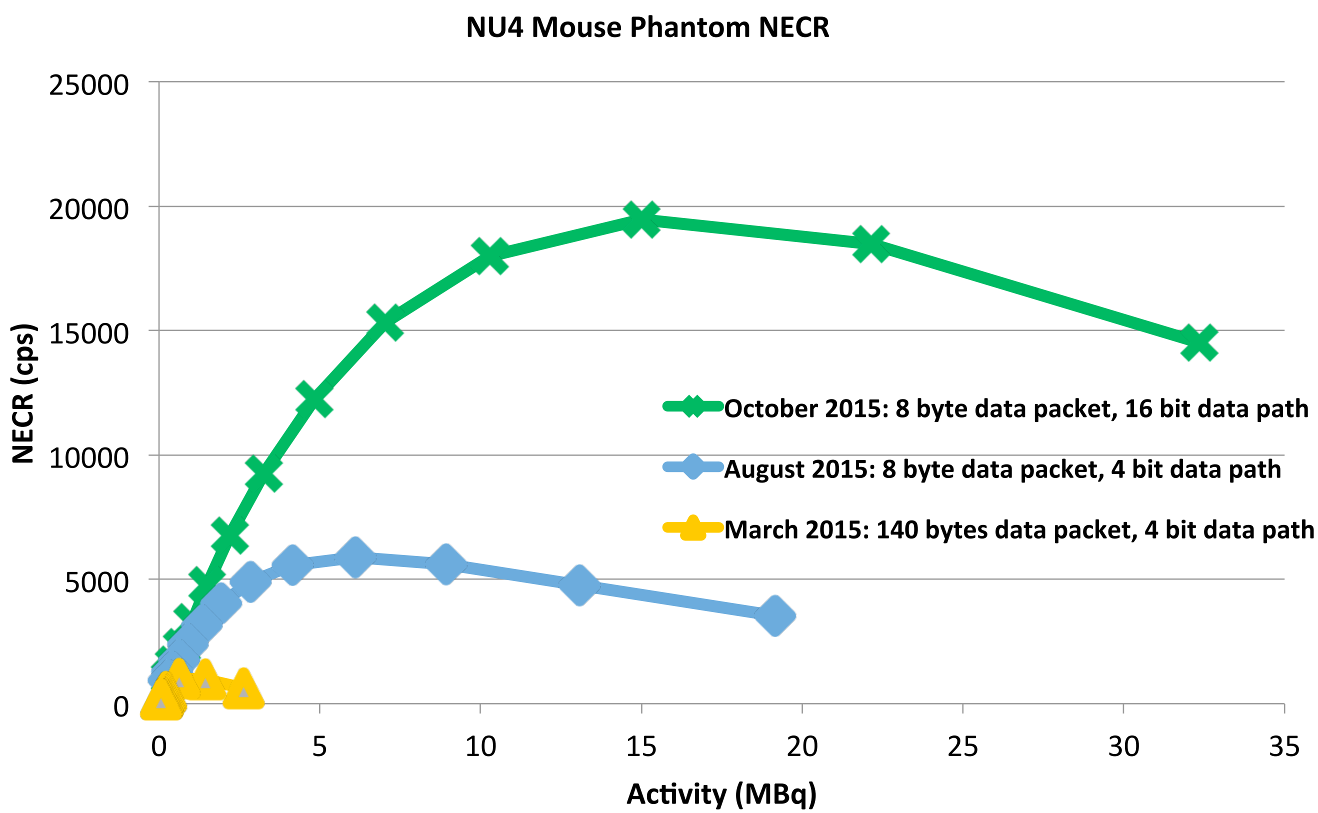




System coincidence event rates measured with a ^{68}Ge line source with ~10 MBq. Based on these results a 4 tick (i.e. 10 ns) coincidence window is used for all acquisitions.

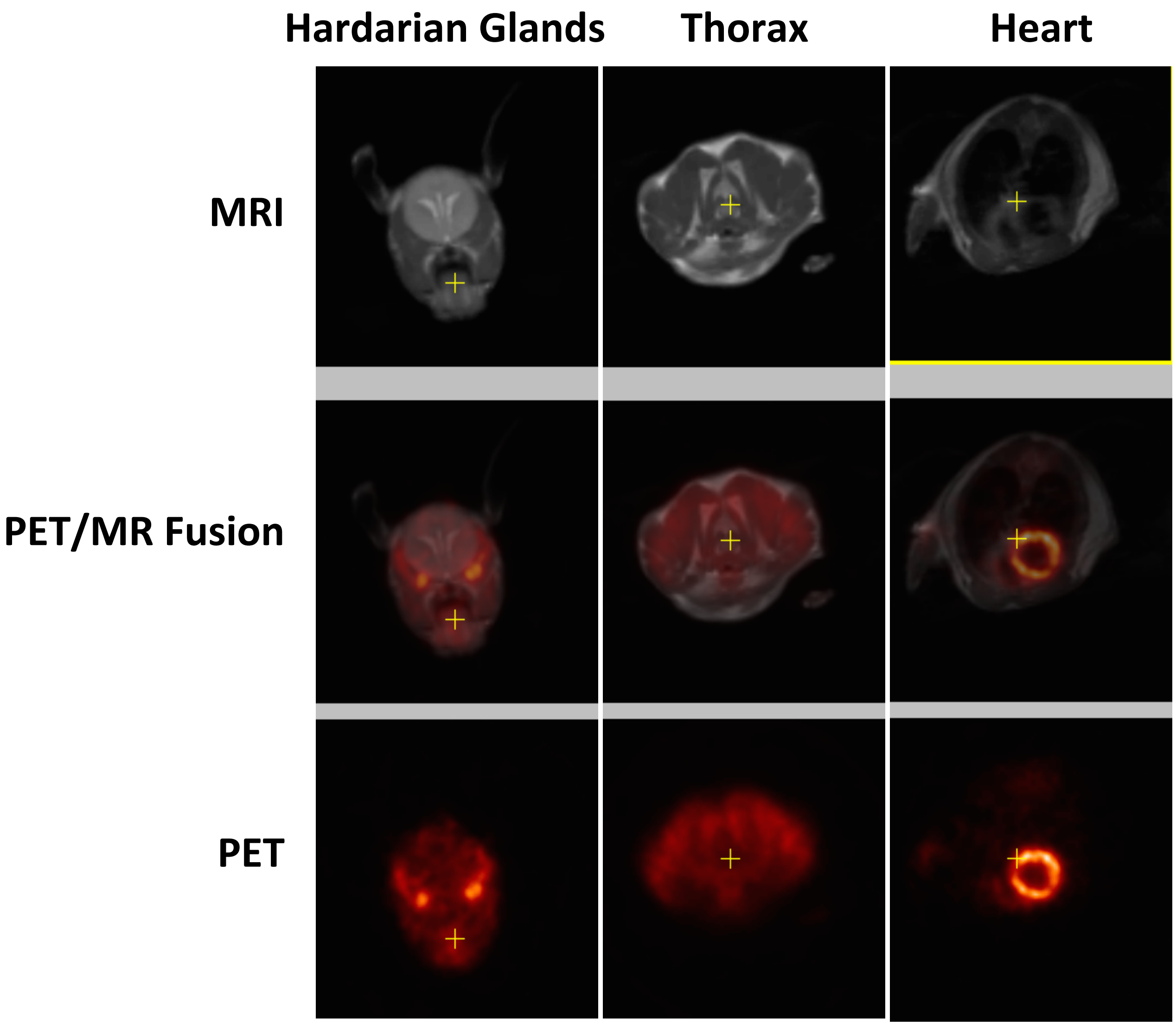
Firmware changes	Upgrades
March 2015	• Time stamp was taken from falling edge of slow signal
October 2015	• The hard reset button used to synchronously reset TDC of all DBs • Fast timing discriminator is used to capture timing from leading edge of fast signal • Independent triggering of detector board and independent data path implemented

Results: NECR gains



- Using this modified firmware, we achieved peak NECR of 19.5kcps @ 15.3MBq

Mouse images



Detailed discussion will be in talk M6B2-7 (High Resolution and Preclinical Systems II: "*First Simultaneous Images and Characterization of a Small Animal PET Insert for PET/MR Imaging*" by G. Stortz.

Summary

- We successfully used OpenPET system to acquire data from 16 detectors simultaneously.
- Using this modified firmware, we achieved peak NECR of 19.5kcps @ 15.3MBq.
- The average energy resolution is 12.5% for the OpenPET acquisition.
- We are now studying the effect of energy DAC threshold value on the system performance

Future enhancements

- We are now limited by USB 2.0 bandwidth (40MB/s)
  - Gbe on host PC interface board needs to be implemented in firmware (hardware existed in PC interface board)
- Global reset of TDC counters through software
- Implement SRAM for event buffering
- Implement energy and crystal lookup tables in FPGA
- Implement onboard coincidence processor

Acknowledgements

This work was funded by the University of Manitoba Amalgamated Research Fund, NSERC Discovery Grants to ALG, CJT and VS, the Faculty of Medicine at the University of Manitoba, and a Mitacs Accelerate Cluster Grant to ALG and VS.

OpenPET with C-SPECT

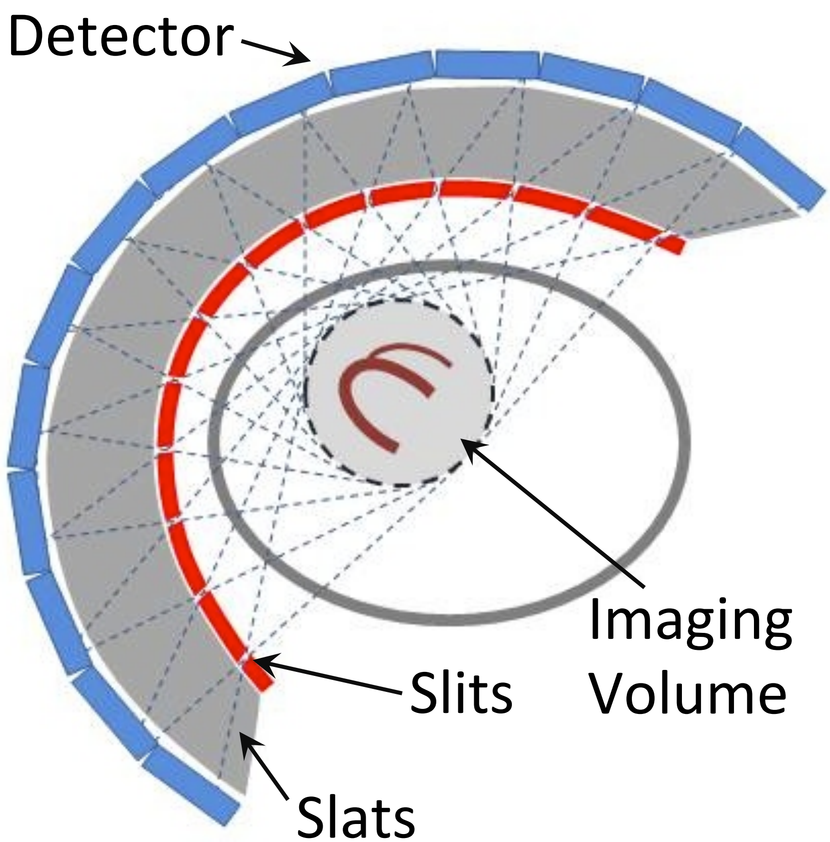
Dale Stentz, Roger Arseneau,
Michael Rozler, Kosta Popovic,
Sankar Poopalasingam, & Wei Chang

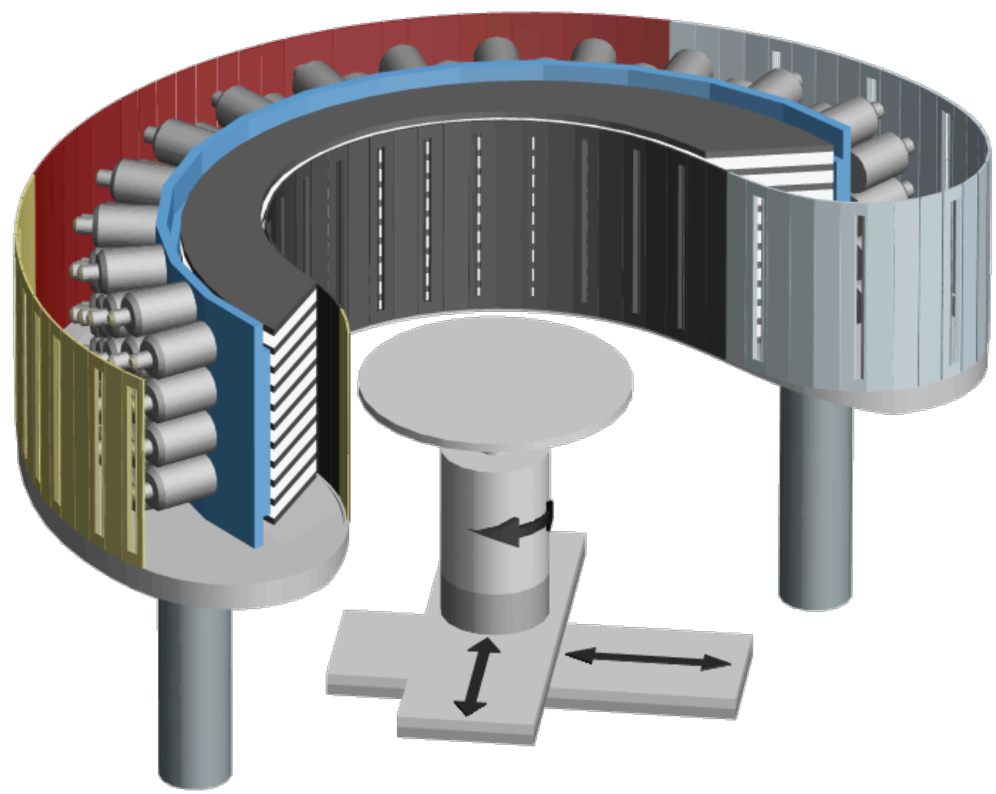
RUSH UNIVERSITY

Overview of Project

- C-SPECT: dedicated cardiac SPECT system for MPI
- Stationary detector designed to maximize system geometric efficiency
- Comprised of 14 detector modules using pixelated NaI (Tl) with 2.5x3.0 mm pixels
- Read out with 13 (2 inch) PMTs per module
- Variable collimation (slit-slat) for different imaging volumes and configurations (e.g. TCT & dynamic imaging)
- Currently have a three module system for testing

Basic Project Design



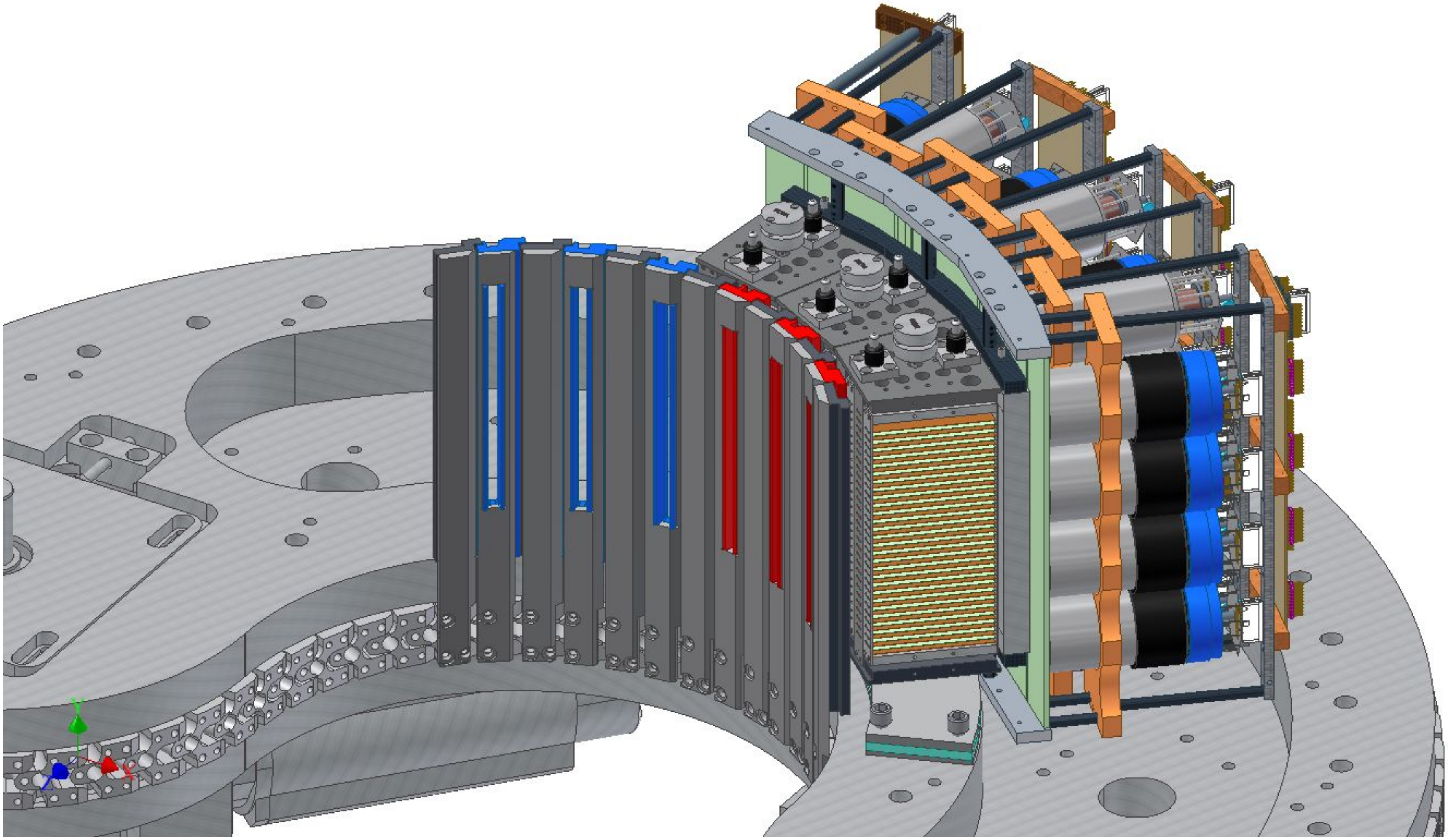


First full lab-prototype designed – see M4D1-1

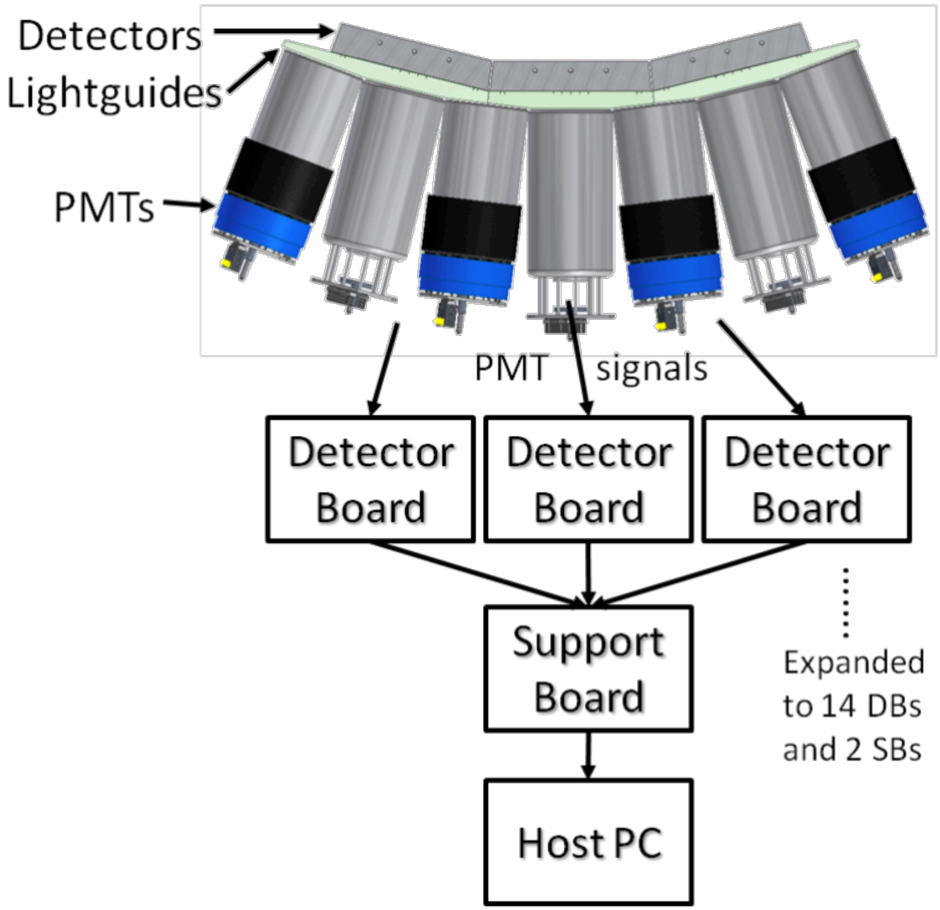
Motivation for OpenPET Use

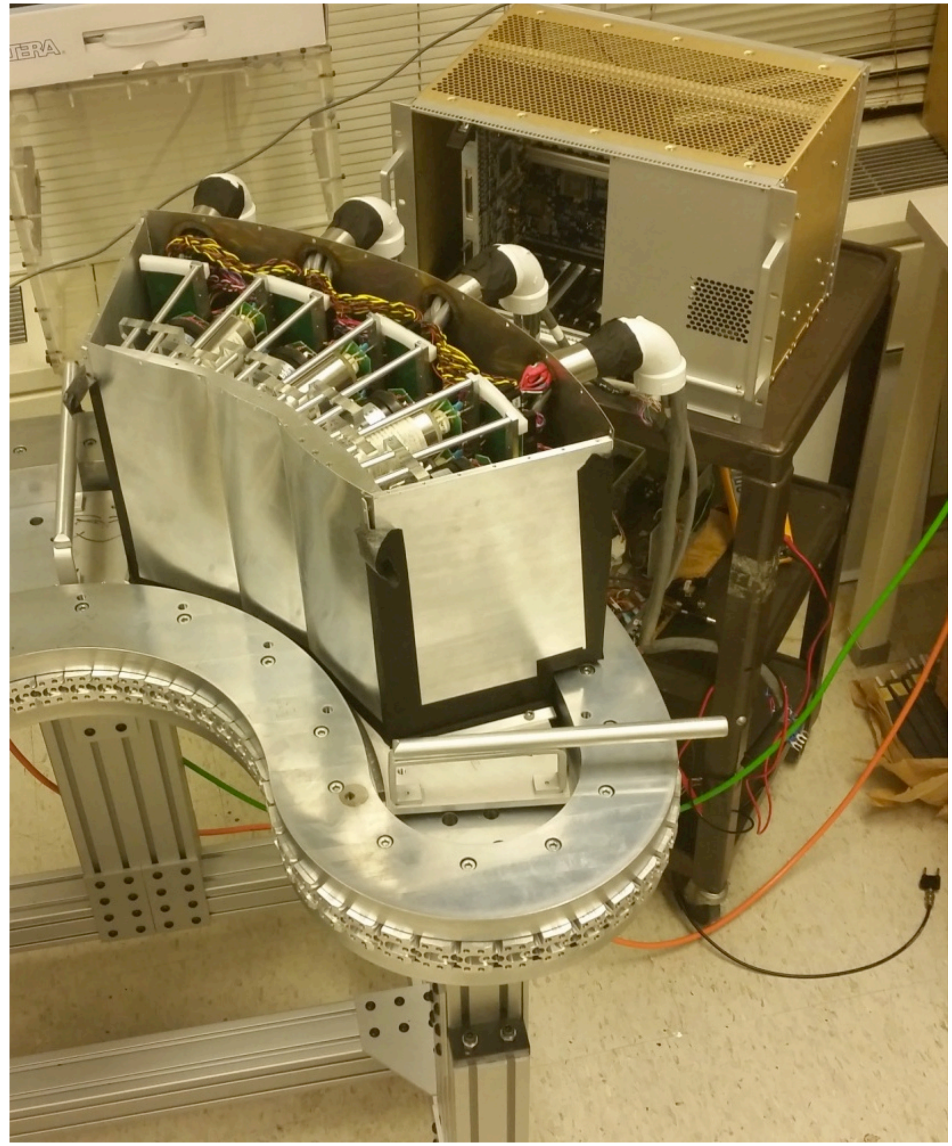
- Scalable/modular DAQ system ideal for Research & Design for the lab prototype C-SPECT system
  - Parallel operations on detector board with each correspond to a detector module
  - FPGA allows for programing of additional “online” features for data acquisition (e.g. pileup correction, light response function correction, etc.)
- Fast with good timing and able to handle high count rates (40 MHz 12-bit ADCs)

Three Module Prototype



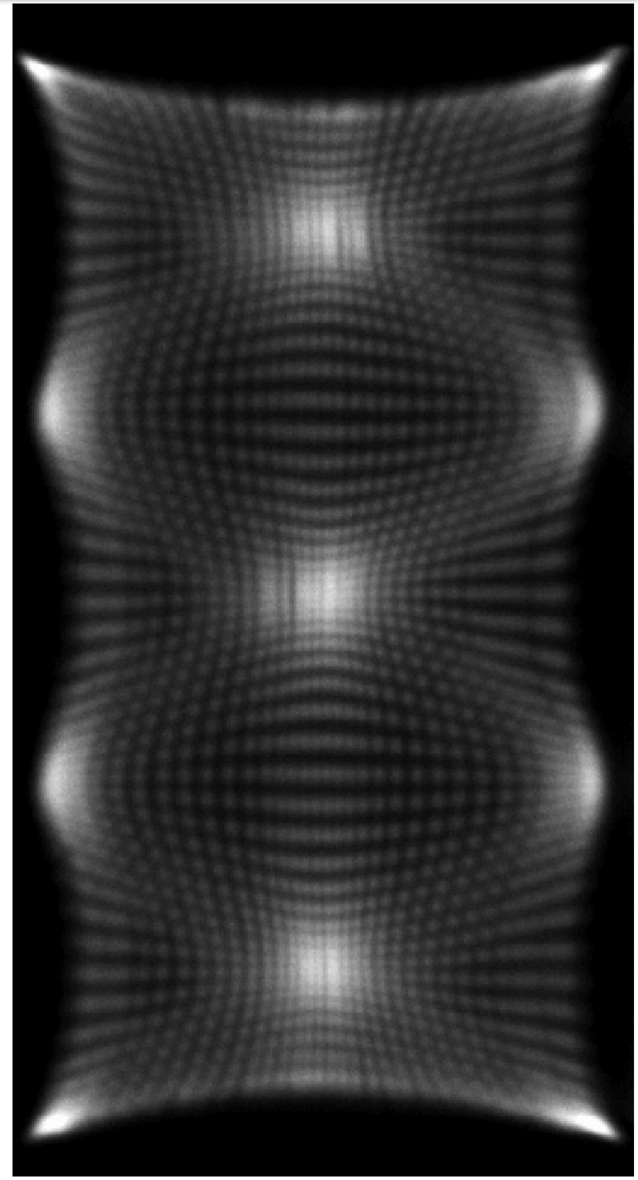
OpenPET with 3 Module System





Three Module Prototype Status

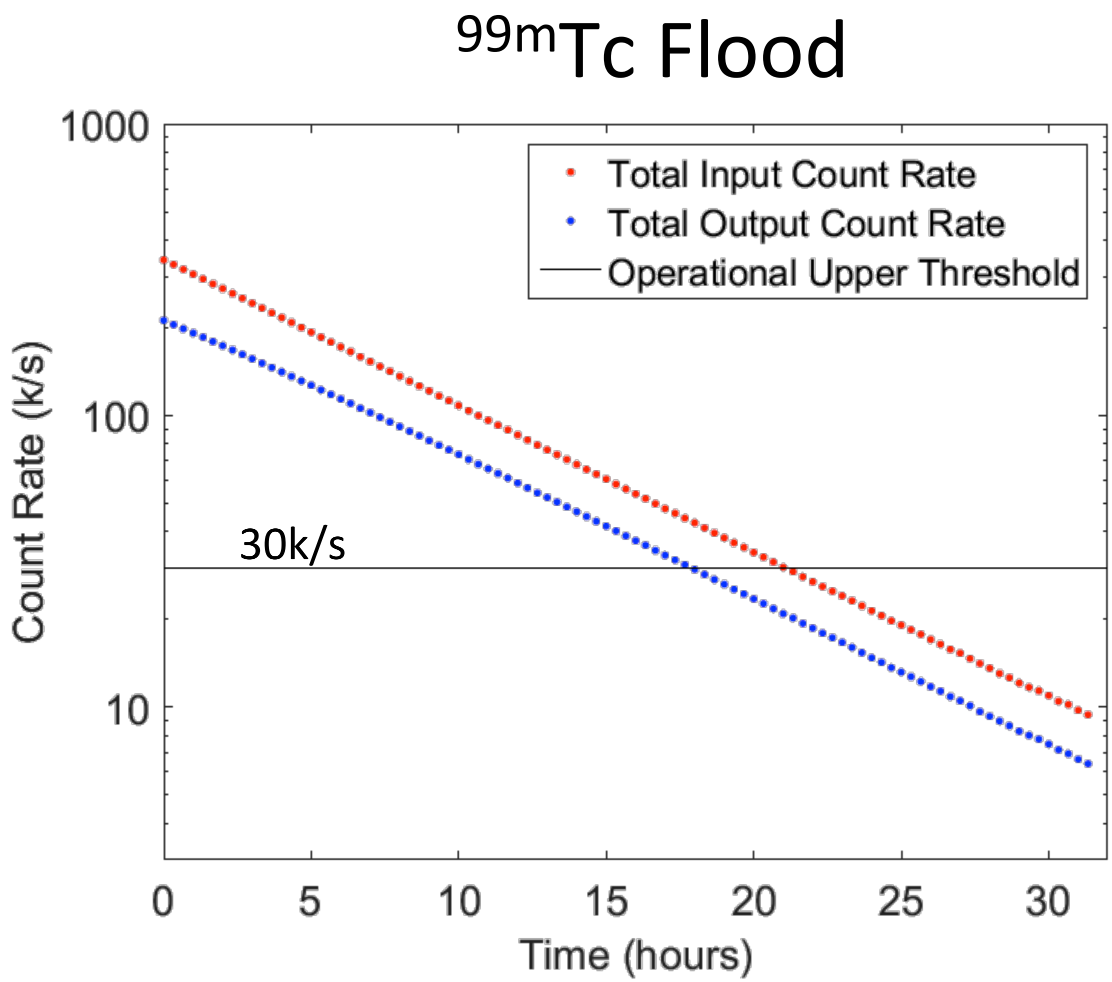
- PMTs well tuned
- Capable of cycling through different collimator configurations for different imaging volumes or functions (TCT, Scout imaging, etc.)
- Full pixel assignment
- System calibration in progress with image reconstruction to follow



OpenPET Use with the Prototype

- Independent support & detector board code
- Acquires data from each detector module with each event word containing the energy at each PMT as well as count rate information
- Ability to apply
  - Full pileup correction (not fully implemented yet)
  - LRF (light response function) correction
  - Linearity correction (position dependent corrections)
- Pulser to take data at low count rates

Total Input Count Rate



- Scan of a flood every 20 minutes over 32 hours
- Input Count Rate: raw 3 module rate
- Output Count Rate: low energy cut & module exclusivity requirement
- We take our emission data with Input < 30k/s
- Work in progress: formal Pile-up correction & evaluation of performance at higher count rates

Future of OpenPET with C-SPECT

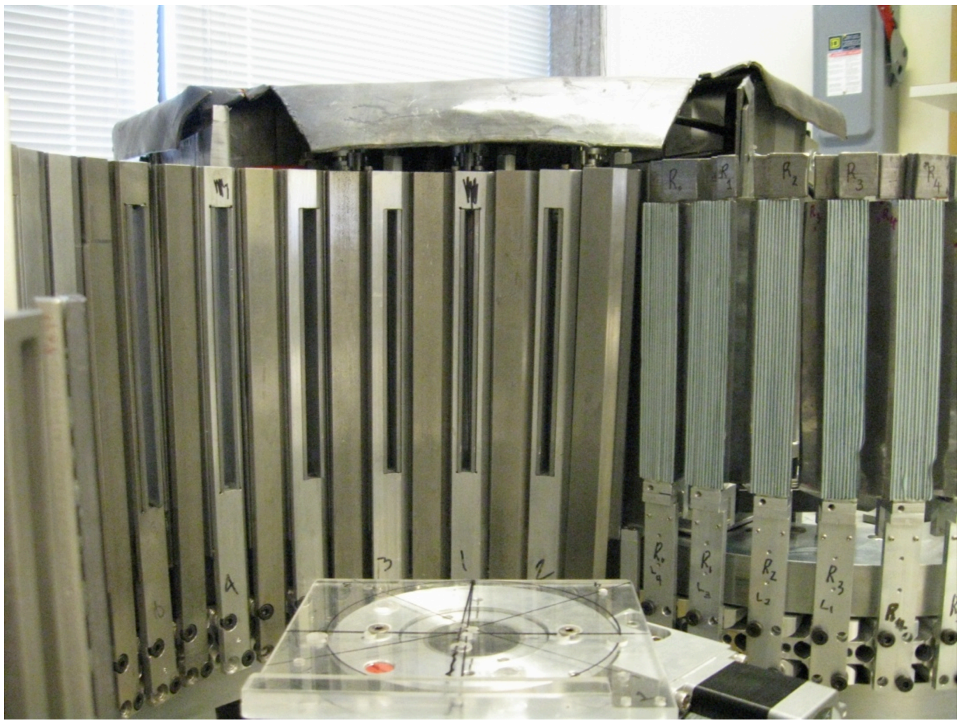
- OpenPET filled our critical need for a DAQ system
- Plan to continue with OpenPET for the full 14 module system

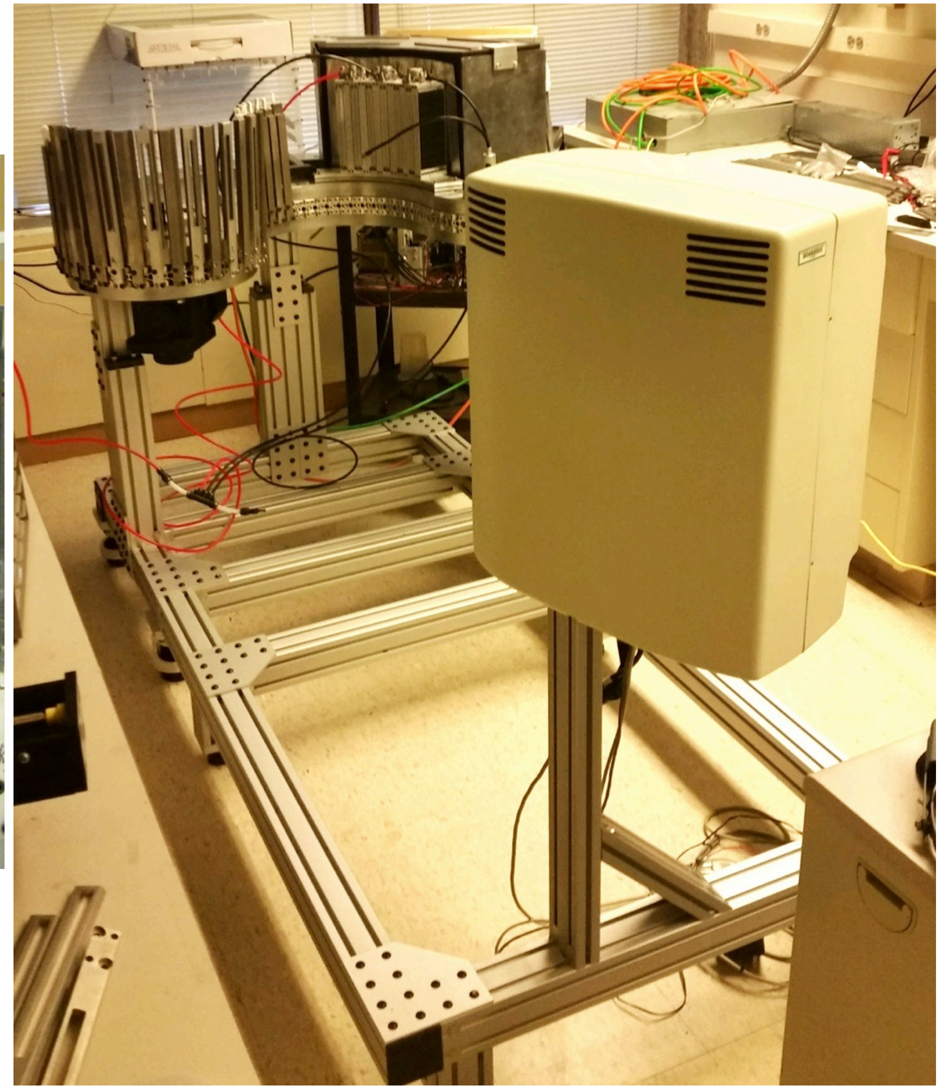
Acknowledgements:

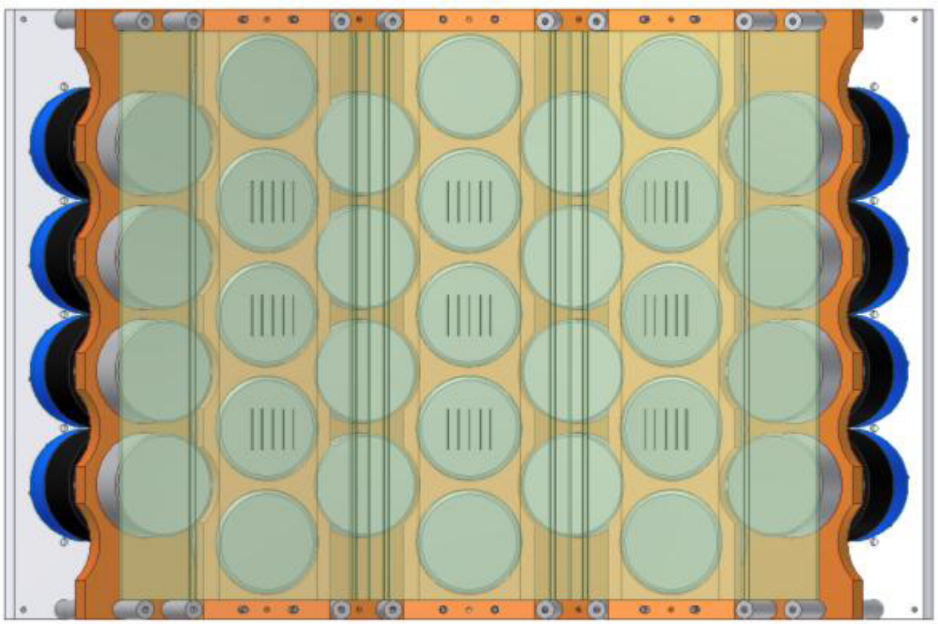
- Thanks to OpenPET collaboration
- Special thanks to Roger for his work on the systems FPGA code as well as for input into the OpenPET development

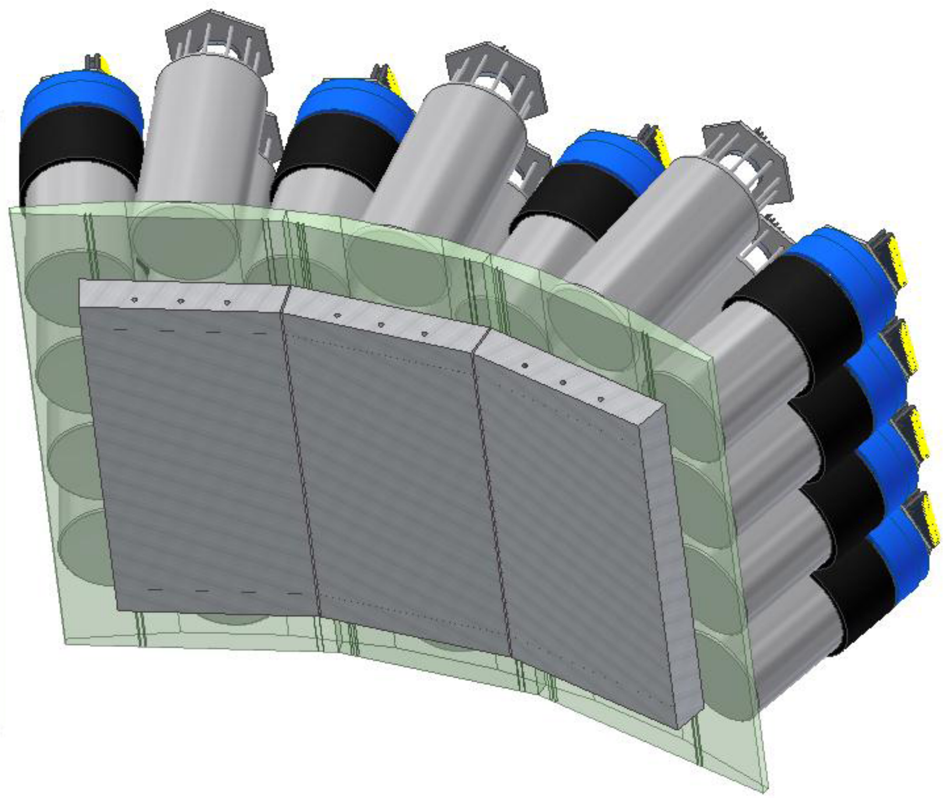
Questions?

Backup Slides

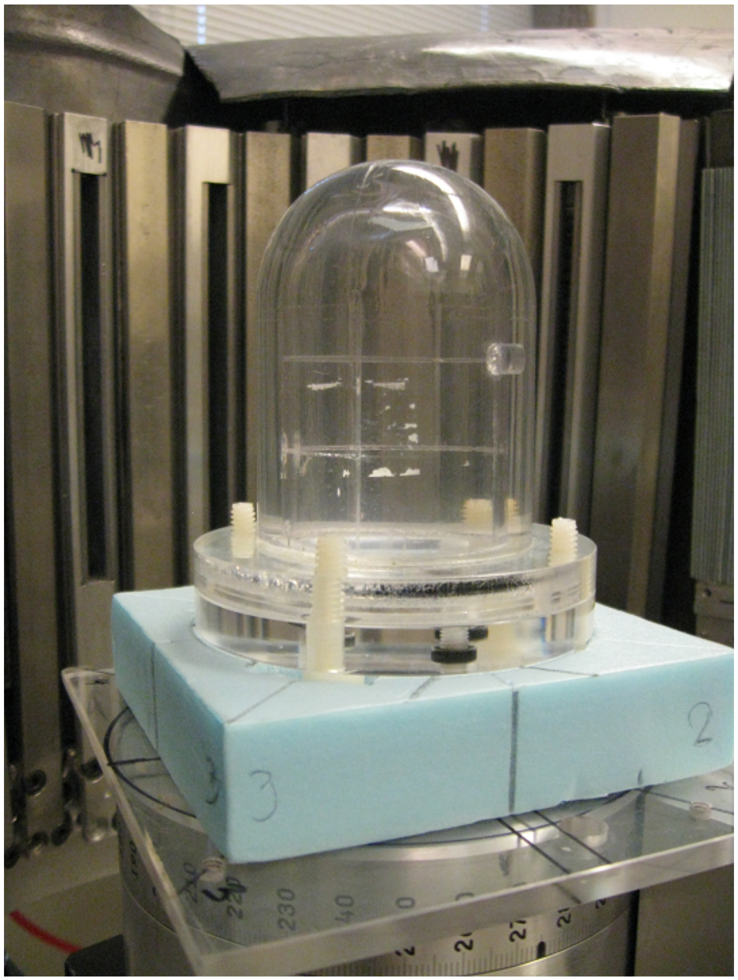


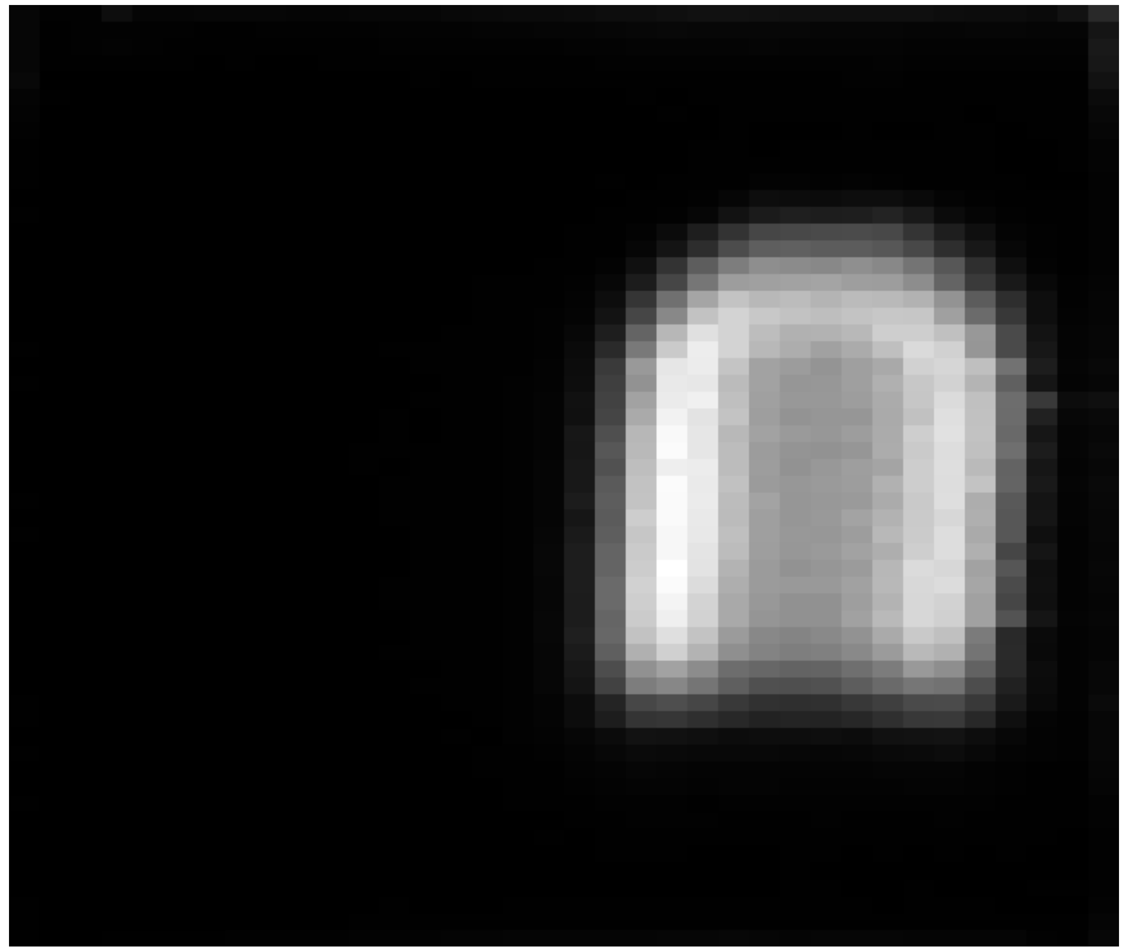






Cardiac Insert Projection





BERKELEY LAB

LAWRENCE BERKELEY NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

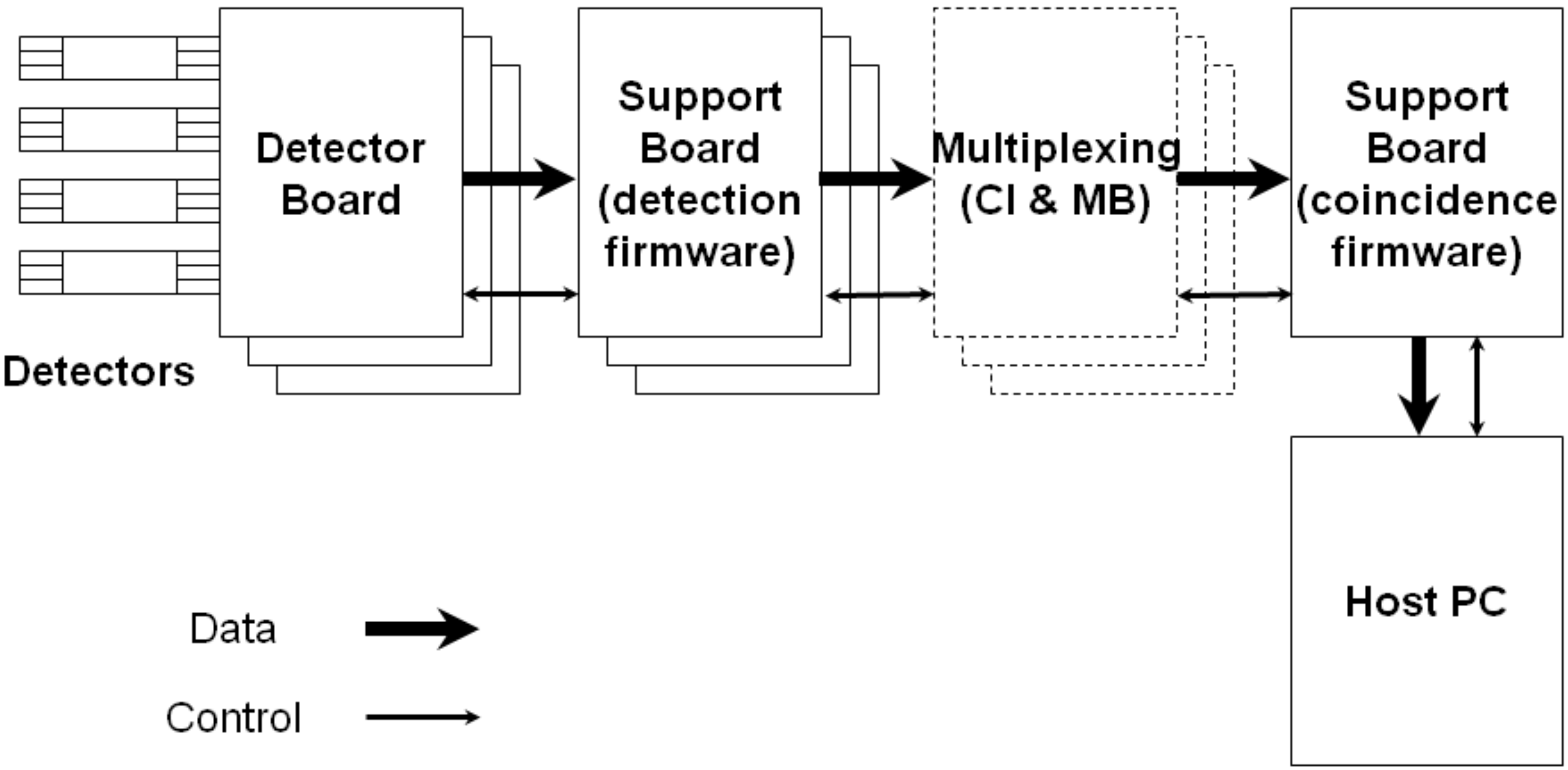
OpenPET User Meeting: Status and Update

Woon-Seng Choong

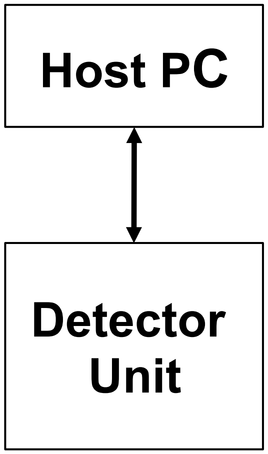
November 5, 2015

This work is supported in part by the Director, Office of Science, Office of Biological and Environmental Research, Biological Systems Science Division of the U.S. Department of Energy under Contract No. DE-AC02-05CH1231 and in part by the National Institutes of Health under grant R01 EB016104.

OpenPET Hardware Architecture

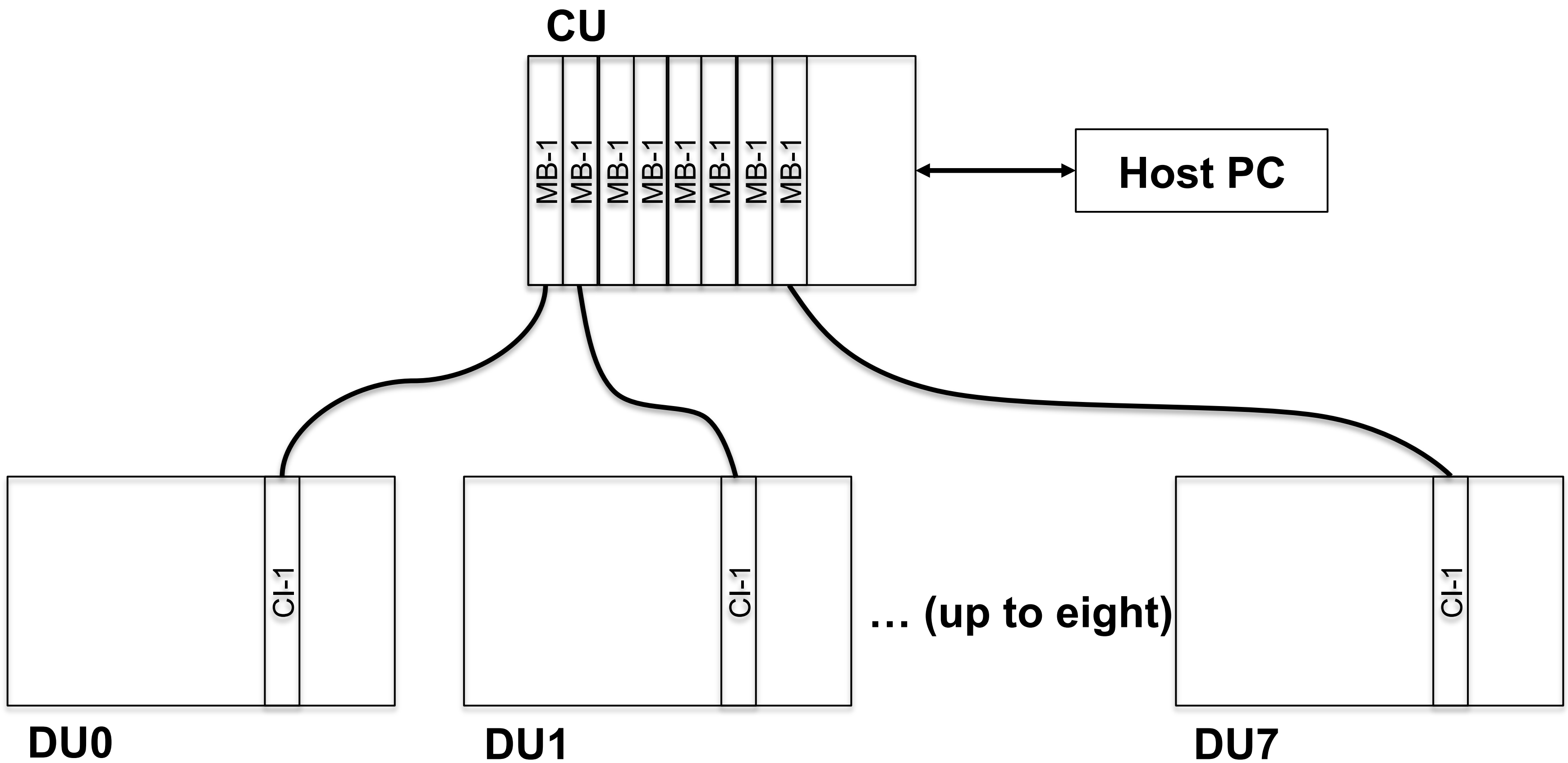


Small System



- 1 Support Crate, Up To 8 Detector Boards
- Up to 256 Analog Inputs (64 Block Detectors)
- PC Interface Board Connects to PC

Standard System



- Up To 8 Detector Units, 1 Coincidence Units
- Up to 2048 Analog Inputs (512 Block Detectors)
- Coincidence Interface Board Connects to CU

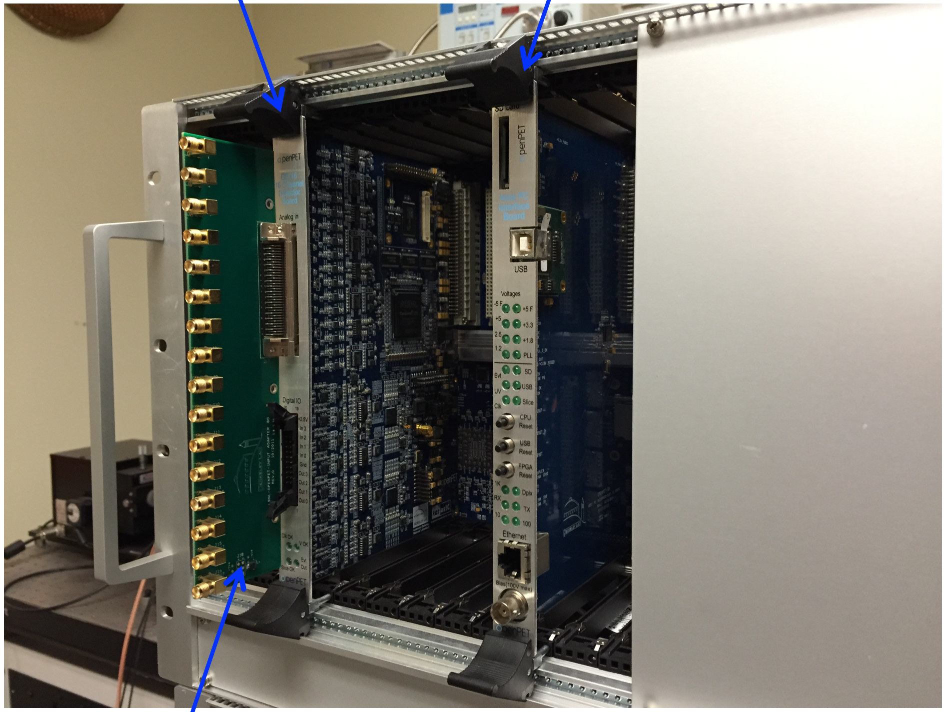
OpenPET Hardware: Detector Unit

Support Board

VME chassis

16-channel Detector Board

Host PC Interface Board



Customized Input Adapter Board

OpenPET Hardware: Getting Started

For a minimum small system

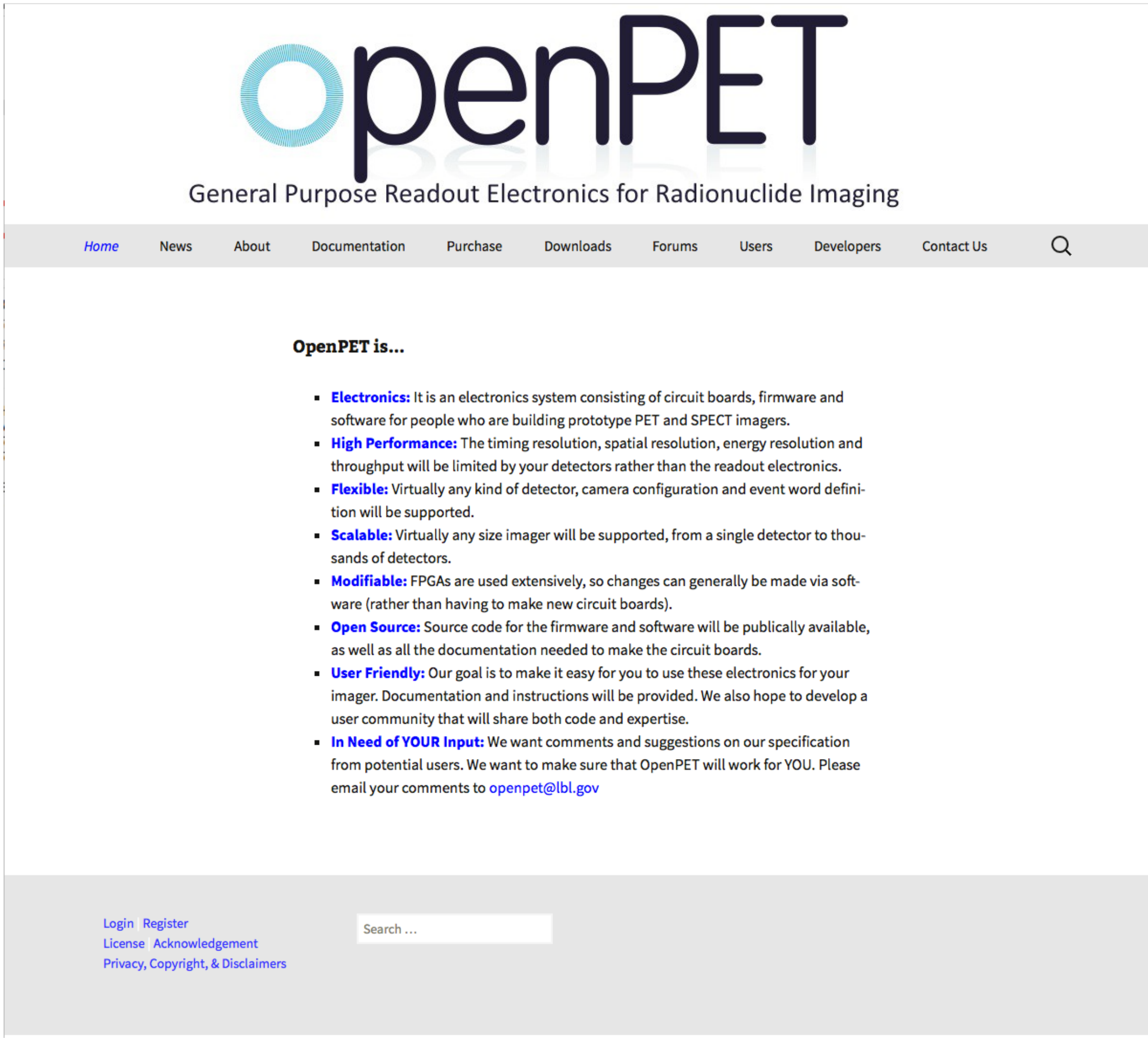
- One Support Board.
- One 16-Channel Detector Board
- One Host PC Interface Board
- One VME chassis

Additional required hardware

- QuickUSB module: Bitwise Systems, part number QUSB2
- USB-Blaster Cable: Terasic, Digi-Key part number P0302-ND

- **All electronics boards can be purchased through Terasic (http://www.openpet.terasic.com)**
- **VME chassis can be purchased through Elma.**

OpenPET Website (openpet.lbl.gov)



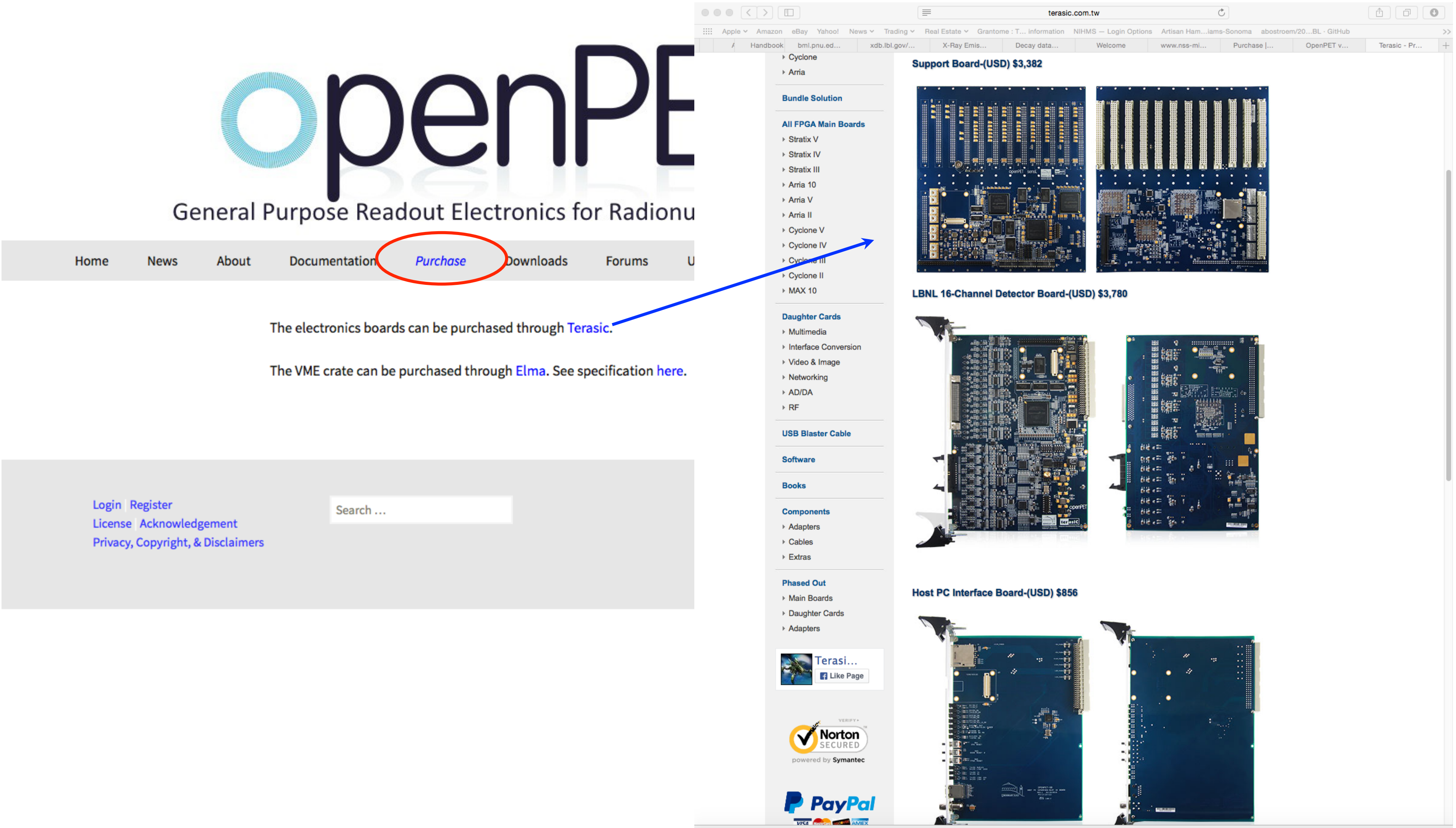
OpenPET is...

- **Electronics:** It is an electronics system consisting of circuit boards, firmware and software for people who are building prototype PET and SPECT imagers.
- **High Performance:** The timing resolution, spatial resolution, energy resolution and throughput will be limited by your detectors rather than the readout electronics.
- **Flexible:** Virtually any kind of detector, camera configuration and event word definition will be supported.
- **Scalable:** Virtually any size imager will be supported, from a single detector to thousands of detectors.
- **Modifiable:** FPGAs are used extensively, so changes can generally be made via software (rather than having to make new circuit boards).
- **Open Source:** Source code for the firmware and software will be publically available, as well as all the documentation needed to make the circuit boards.
- **User Friendly:** Our goal is to make it easy for you to use these electronics for your imager. Documentation and instructions will be provided. We also hope to develop a user community that will share both code and expertise.
- **In Need of YOUR Input:** We want comments and suggestions on our specification from potential users. We want to make sure that OpenPET will work for YOU. Please email your comments to openpet@lbl.gov

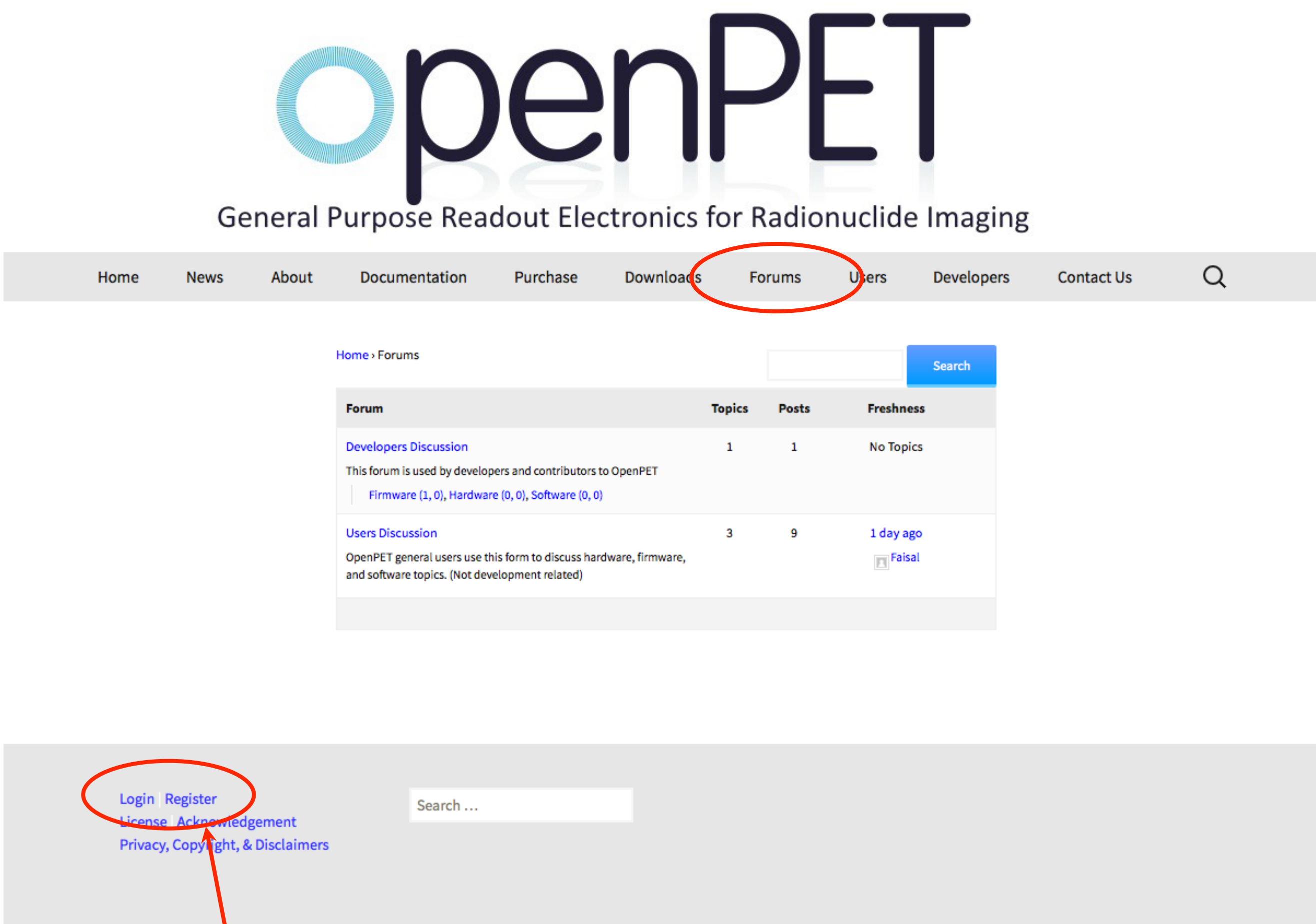
Login Register
License Acknowledgement
Privacy, Copyright, & Disclaimers

Search …

OpenPET Website (openpet.lbl.gov)

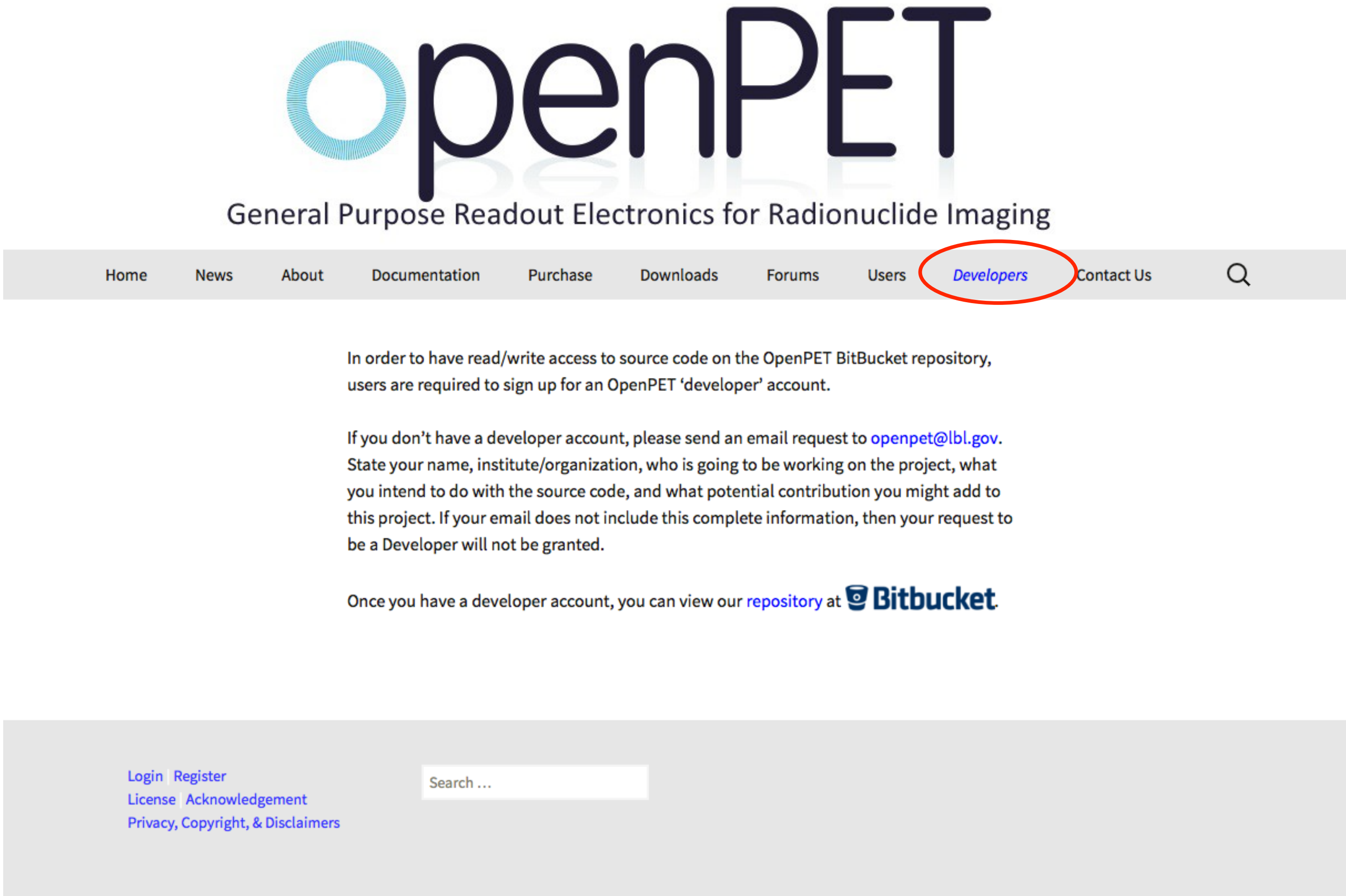


OpenPET Website (openpet.lbl.gov)

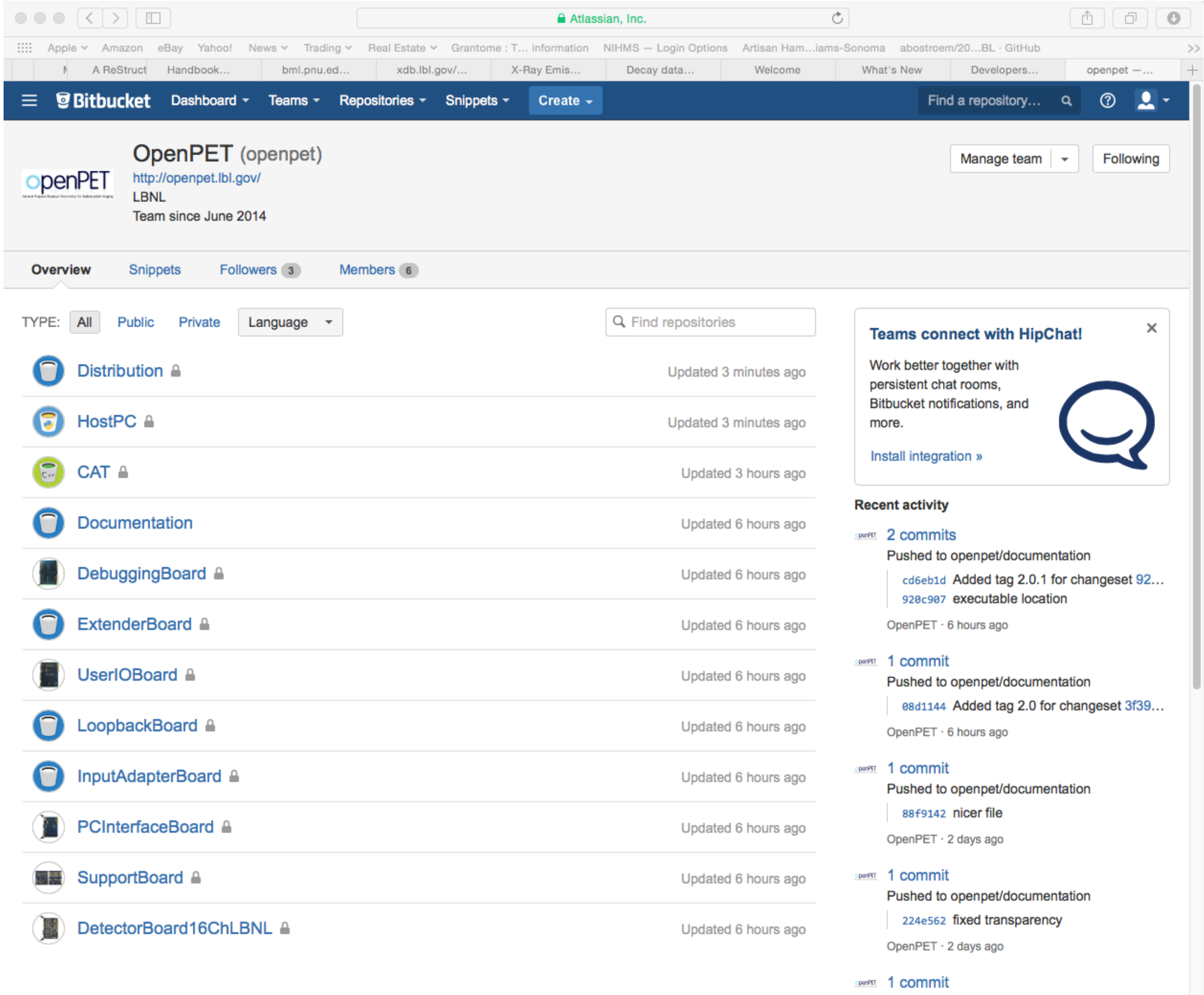


Must be registered and login to post and reply

OpenPET Website (openpet.lbl.gov)

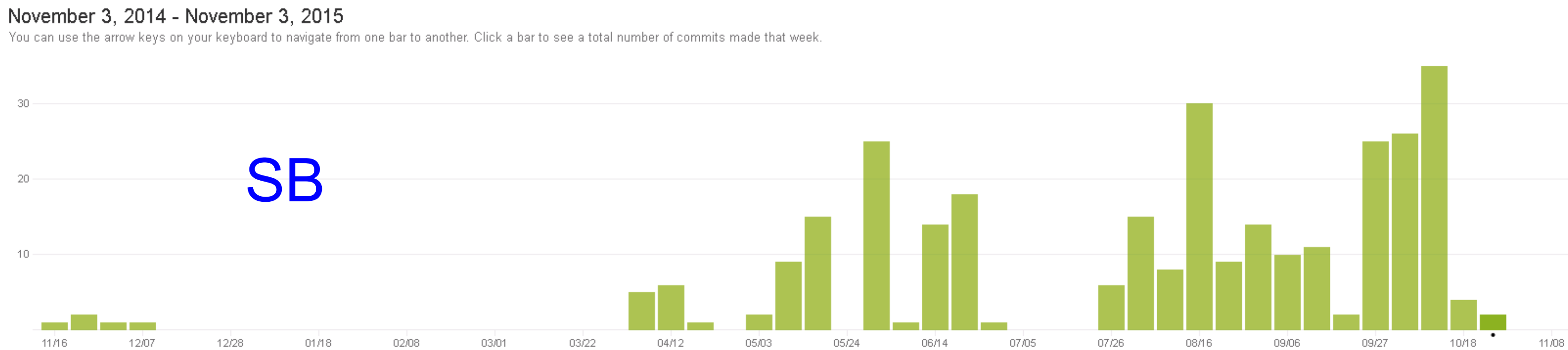


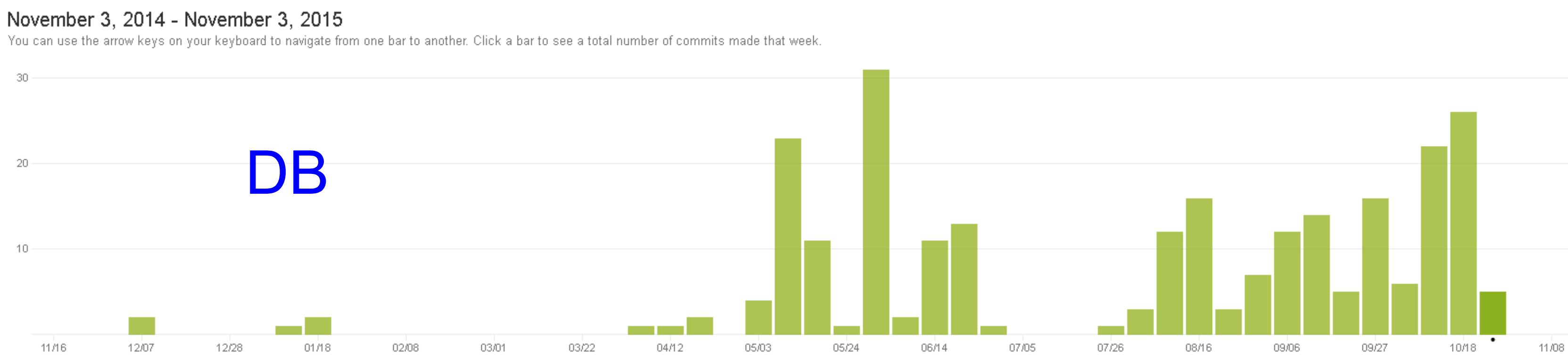
Bitbucket Repository (https://bitbucket.org/openpet/)

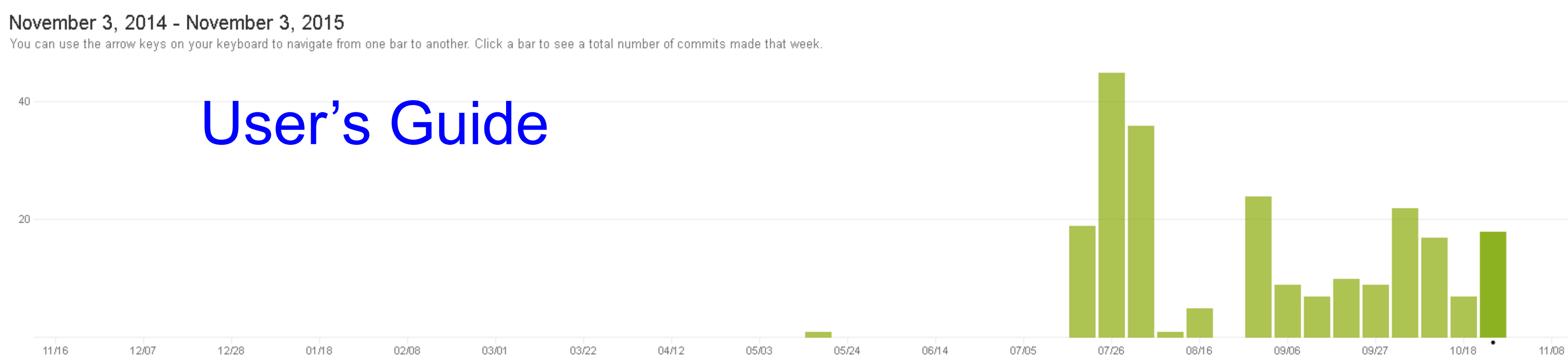


- Has to be approved as a developer to access the repository (except the Documentation repository)

Recent Repository Commits







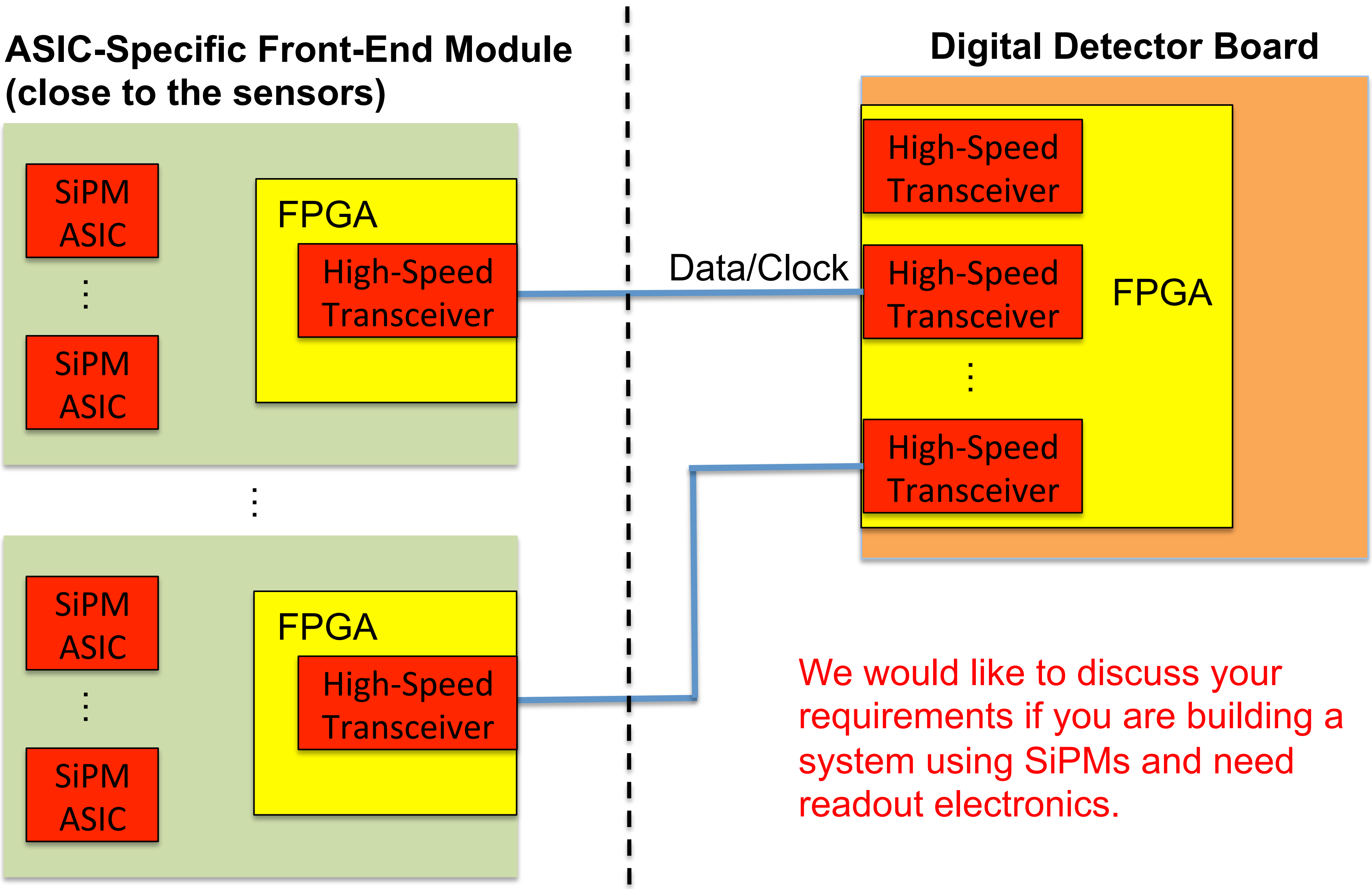
Current State of Hardware and Firmware

- Hardware:
  - SB and 16-channel DB have been evaluated and are working as designed.
  - 32-channel DB has been fabricated, but has not been evaluated.
  - Multiplexer Board-1 has been fabricated to enable the Standard System.
  - QuickUSB on PC Interface Board works as expected.
- Firmware:
  - Completed a rewrite of the firmware and software architecture, dubbed OpenPET v2.0 has been released:
    - ✓ Well-defined architecture.
    - ✓ Improves the organization, usability, stability, robustness, etc to ensure a solid foundation for subsequent development.
    - ✓ High-resolution TDC has been implemented.
    - ✓ Interface to SRAM has been implemented.
    - ✓ 128-bit (100 ns) or 256-bit (200 ns) data word for real-time processing has been implemented.

Current State of OpenPET Software

- Software:
  - Low level.
    - Scripting Interface to automate configuration, , acquisition, and control.
    - OpenPET library for direct interfacing with system (Python, C/C+).
    - Real-time software processing using Python (NumPy) and C/C++.
    - High performance queuing for optimal real-time acquisition.
    - Supports multiple operating systems Windows, Mac, and GNU/Linux.
  - High level
    - Limited user interface, diagnostic, and calibration software.

Digital Detector Board



We would like to discuss your requirements if you are building a system using SiPMs and need readout electronics.

Future Work

- Hardware:
  - Digital Detector Board is being designed to interface with custom front-end readout IC (SiPM) that output digital signals.
  - Evaluate the 32-channel Detector Board.
- Firmware:
  - GbE on PC Interface Board is being implemented.
  - Enable the Standard System.
  - Develop event processing modules. This task is application-specific and we welcome collaboration.

OpenPET Lab @ LBNL

- Test and evaluate OpenPET electronics before committing.
- In-person or remote desktop login.
- Bring/send your own detector module or use in-house detector module.
- We welcome collaboration.

BERKELEY LAB

LAWRENCE BERKELEY NATIONAL LABORATORY

U.S. DEPARTMENT OF ENERGY

OpenPET v2.0

Faisal Abu-Nimeh, Ph.D.

20151104

What's New?

- No changes on hardware.
- Firmware, Embedded Software, and Software:
  - Complete rewrite from scratch.
  - No failed timing closures.
  - Generic ADC interface. Supports multiple TI ICs.
  - Modular Modes: IDLE, SCOPE, SINGLES, …
  - Modular Actions: START, STOP, RESET, …
  - Modular 32-bit settings for a given Mode.
  - Unified firmware architecture on all FPGAs.
  - Scripting Interface and SDK.
  - Standardize all implementations e.g. SPI.
  - All configurations are parameterized to allow customization, e.g. FIFO depths, TDC, SINGLES pipeline, etc.
  - New software interface with multiprocessing, queuing, and threading.
  - MANY MORE, see:
    - http://openpet.readthedocs.org/en/2.0.1/release_notes.html#what-s-new

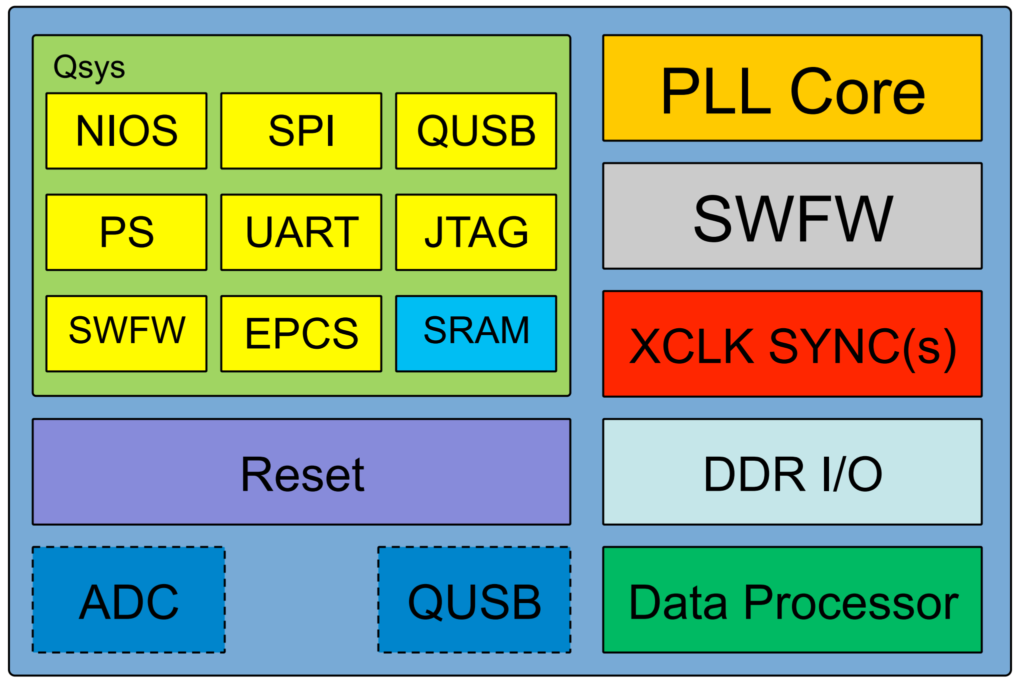
Firmware Changes I

- Unified architecture on all FPGAs:
  - Minimalistic NIOS II Microprocessor on all FPGAs.
  - Same clocking; PLL instance is the same on all FPGAs.
  - Standard 32-bit SPI interface between all nodes.
  - Standard Software-Firmware interface on all FPGAs.
  - Standard Double Data Rate LVDS transceiver on all FPGAs for high-speed data transfers.
  - Scope and Singles data processing cores on all.

Unified Firmware Architecture

Main, IOs, and DB FPGAs share the same architecture.

This allows us to reuse the same blocks in all FPGAs which yields to more stability, faster development, and better readability.

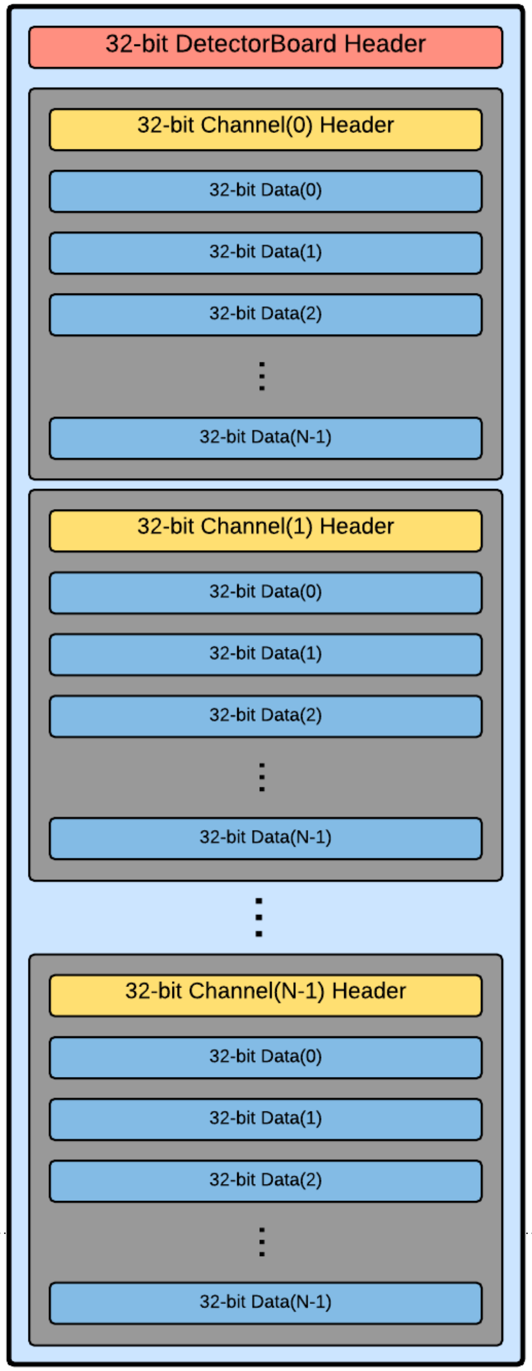


Firmware Changes II

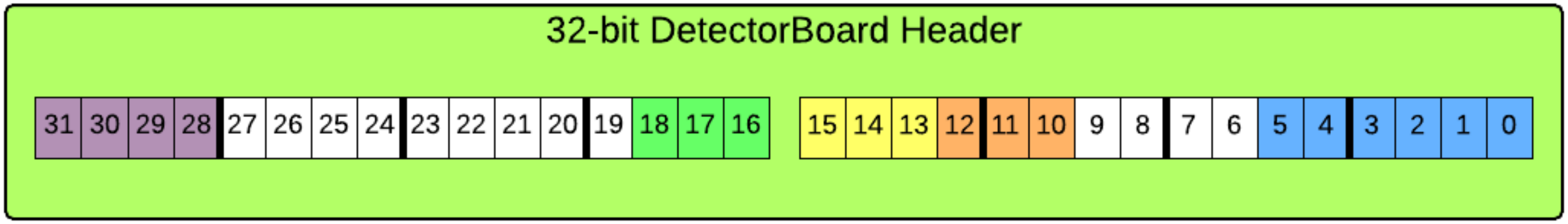
- Modular Modes, Actions, and Settings:
  - 2^{32} possible Modes. Currently we have:
    - Scope Mode: Raw ADC data.
    - Singles Mode: Processed ADC data e.g. energy.
    - Idle Mode: nothing processed or transferred.
  - 2^{32} possible Actions. Currently we have:
    - Run: launches a selected Mode.
    - Stop: causes a selected Mode to pause.
    - Reset: causes a selected Mode to reset to default.
  - 32-bit wide Settings for a given Mode.

Scope Mode – Data

- Sends raw ADC data to external disk storage.
- Uses 16-bit wide bus to transfer raw data.
- Uses DDR to send 32-bits at a time.
  - Each 32-bit packet has a 4-bit packet ID.
- Each DB wraps its data with a DB Header.
- Each channel in a given DB wraps its data with a Channel Header.
- All DBs are synchronized when they start the acquisition.



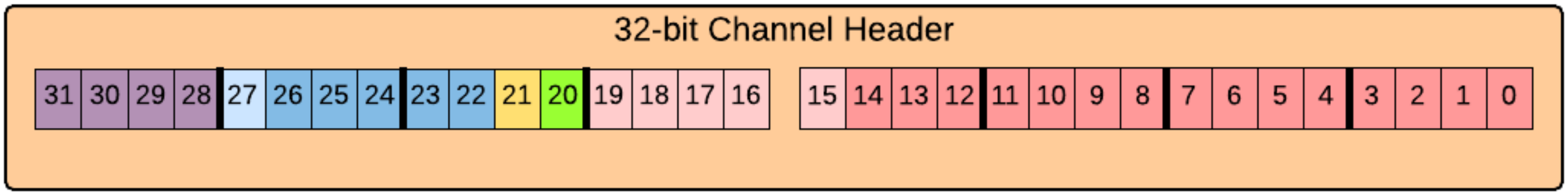
Scope Mode – DB Header Packet



Starting from least significant bit (LSB)

- (5:0) Number of channel header packets (i.e. 0 to 63)
- (9:6) not used
- (12:10) Detector Board Address (populated by parent)
- (15:13) DU Address (populated by parent)
- (18:16) MB Address (populated by parent)
- (27:19) not used
- (31:28) Packet ID (must equal to 0x4)

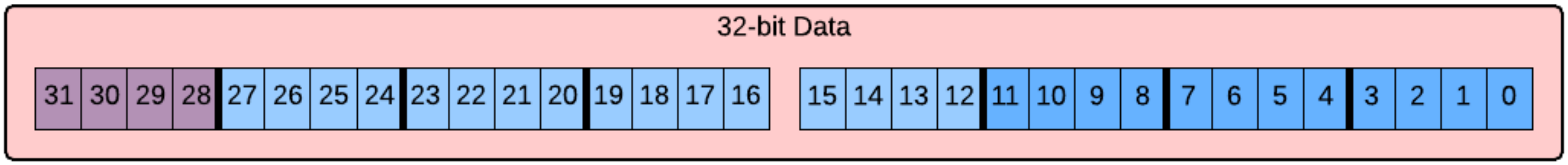
Scope Mode – Channel Header Packet



Starting from least significant bit (LSB)

- (19:0) TDC data (if used)
- (20) hw trigger hit (energy)
- (21) fw trigger hit
- (27:22) channel address (i.e. 0 to 63)
- (31:28) Packet ID (must equal to 0x3)

Scope Mode – Data Packet



Starting from least significant bit (LSB)

- (27:0) e.g. raw ADC data from (11:0)
- (31:28) Packet ID (must equal to 0x1)

Scope Mode – Settings

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Starting from least significant bit (LSB)

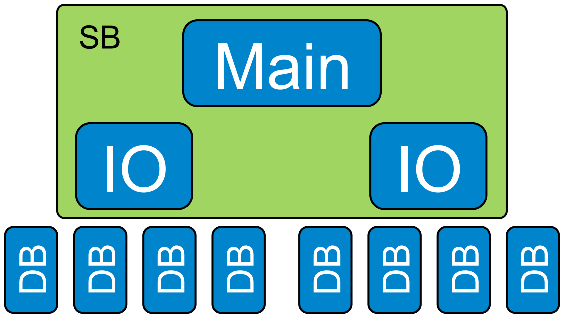
- (3:0) Reserved: Must equal to 0001
- (12:4) Total Number of ADC samples (2^9-1 = 511), (zero is accounted for)
- (15:13) Reserved
- (19:16) Number of ADC samples before energy trigger (2^4-1 = 15)
- (23:20) Reserved
- (27:24) Trigger window (2^4-1 = 15)
- (31:28) Reserved

Notes:

- Total Number of samples should be greater than samples before trigger + trigger window.
- Total Number of samples should not exceed Firmware's maximum number of samples - (Number of Channel headers + Detector Board Header)

Scope Mode – Multiplexing

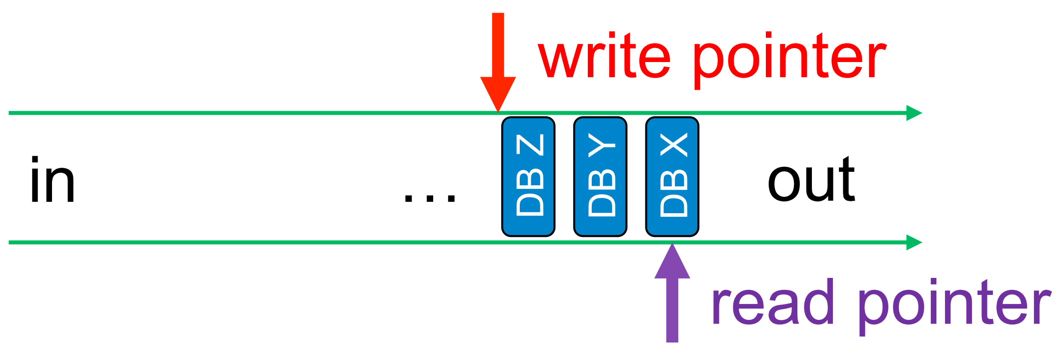
- For a small system, i.e., single chassis:
- Multiplexing occurs in SB at:
- No data loss.
  - IO FPGAs: 4-in-1-out
  - Main FPGA: 2-in-1-out



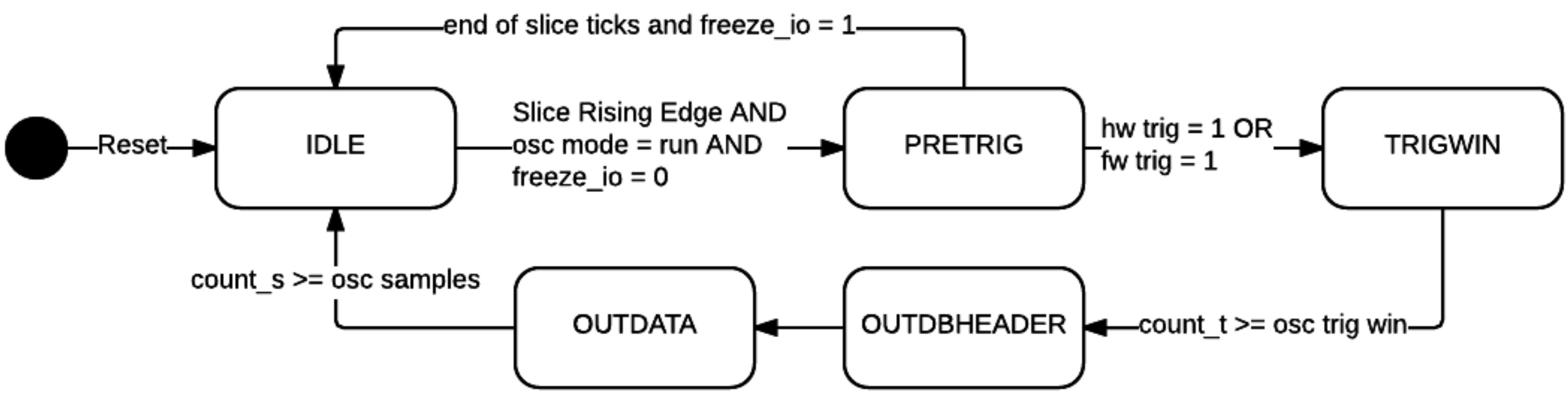
- Two available options for multiplexing:
  - (default) Synchronized data management across all DBs, i.e. aligned readout on all DBs.
    - Fair to random triggers (round-robin).
    - Biased toward periodic triggers.
    - Longer dead time.
  - First-in-first-out scheduler:
    - Fair to periodic triggers.
    - Biased toward DBs with higher trigger rates.
    - Shorter dead time.
- Reusable code on Main and IO FPGAs on SB.

Scope Mode – Queuing

- Queue depth equals 2 * number of inputs:
  - IO FPGA queue depth is 8.
  - Main FPGA queue depth is 4.
- Generic reusable code for Main and IO FPGAs.
- Incoming blocks of data are tagged, write pointer advances.
- Once data is read out, queue slot is emptied and read pointer advances.
- R/W Pointers are circular.



Scope Mode – DB Statemachine

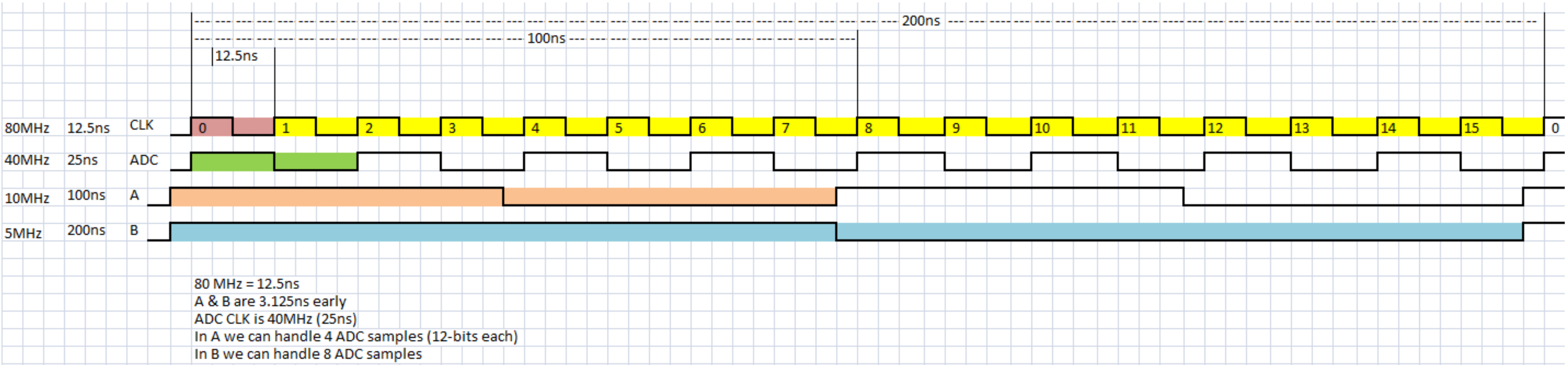


Scope Mode – Binary file format

- Headers are prepended to raw data.
  - Time of Acquisition.
  - All acquisition Settings.
- 4KB are reserved for user defined content.
- Simple 32-bit format for easy parsing and manipulation.

Offset	[31..24]	[23..16]	[15..8]	[7..0]	Comment
0	Epoch timestamp				Local Time
1	Reserved				
2	Acquisition Duration				From Software
3	Acquisition Mode Payload				See cmd id 0x0003
4	Acquisition Mode Settings Payload				See cmd id 0x0005
5	Reserved				future data format
6	Misc.Software Settings				processed/raw?
7					
...	Reserved				
9					
10					
...	User Defined				
999					
B=1000	Detector Board Header				M=Num of channels
B+1	Channel Header (0)				
B+2	ADC Sample (0)				
B+3	ADC Sample (1)				N=number of ADC samples. *
	...				
B+N	ADC Sample (N-1)				
B+1+N	Channel Header (1)				
	...				
	Channel Header (M-1)				
	...				
B+M*(1+N)	Detector Board Header				Next Detector Board
	...				
	Detector Board Header				Next Detector Board
	...				

Singles Mode – Intro



- Two ‘Slice’ widths supported.
  - 128-bit words and 256-bit words.
- Customizable pipeline stages depending processing needs (user or algorithm defined).
- Just like Scope mode, uses DDR to send 32-bits at a time.
  - Each 32-bit packet has a 4-bit packet ID.
- User defined cores are isolated from Singles top level core.

Singles Mode – Packets

R = SystemClkPeriod/ADCClkPeriod

N = R * SlicePeriod/SystemClkPeriod

procs_ticks = (*g_max_pipeline_stages*-1)*N

#	32-bit wide word [31:0]
0	Packet 0
1	Packet 1
...	...
N-1	Packet N-1

Slice choice (1):

Default System Clk Period = 12.5 ns (80MHz)

Default ADC Clk Period = 25 ns (40MHz)

Slice Period = **100 ns** (10MHz)

R = 12.5/25 = 1/2

N = 1/2 * 100/12.5 = **4** packets

Total bits transferred per Slice = 4*32 = **128 bits**

Slice choice (2):

System Clk Period = 12.5 ns (80MHz)

Default ADC Clk Period = 25 ns (40MHz)

Large Slice Period = **200 ns** (5MHz)

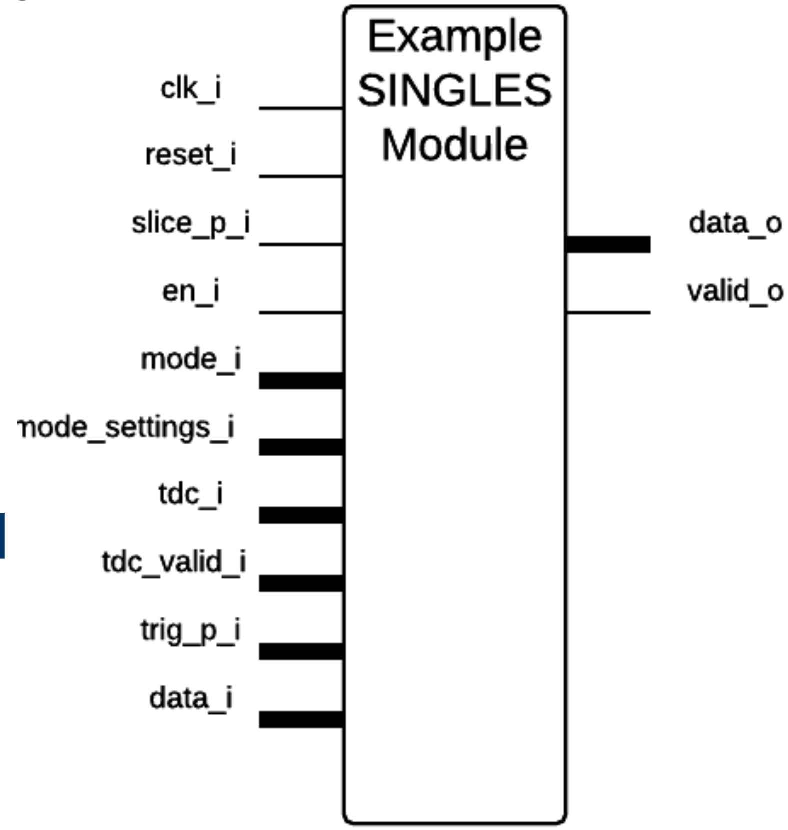
R = 12.5/25 = 1/2

N = 1/2 * 200/12.5 = **8** packets

Total bits transferred per Slice = 8*32 = **256 bits**

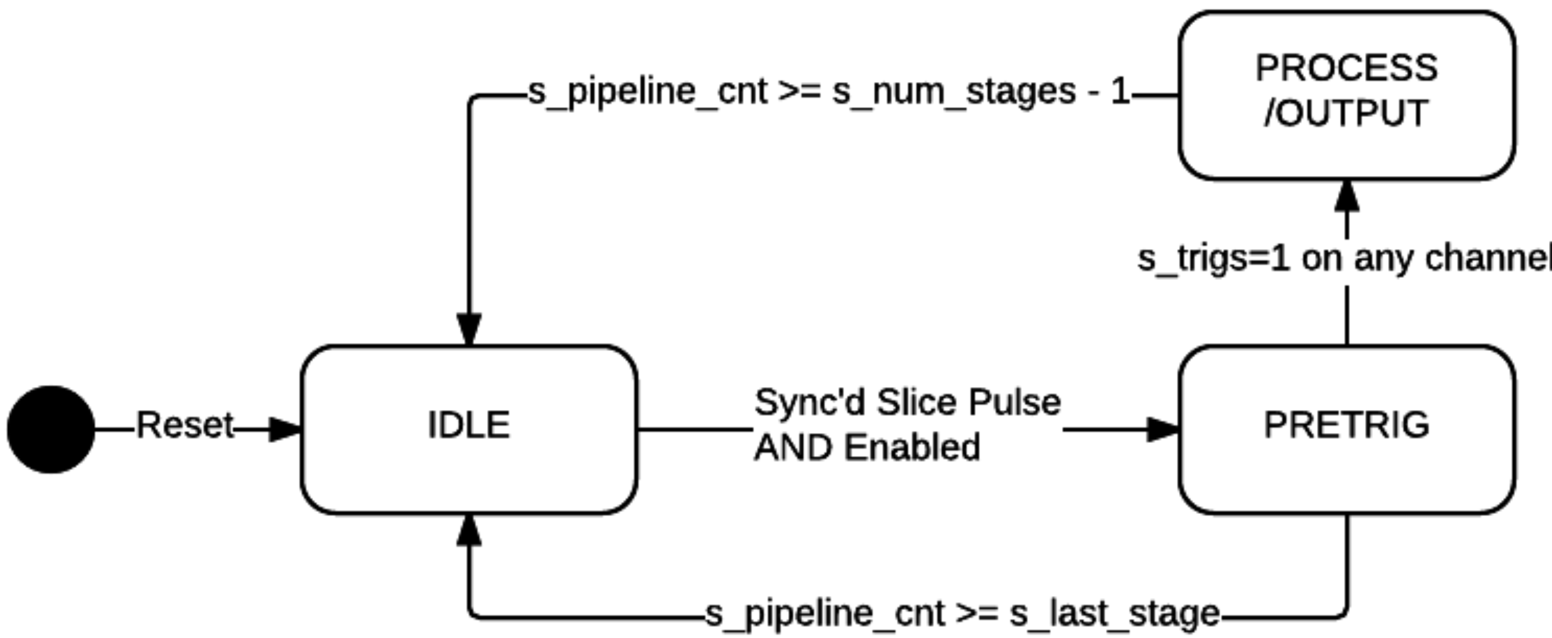
Singles Mode – Interface

- Isolate user-defined cores.
- Provides I/O shown in figure →
  - More I/O can be added.
- Interface intelligently instantiates the correct number of user-defined cores based on the number of pipeline stages available.
- Singles data is handled by the interface without any user intervention.



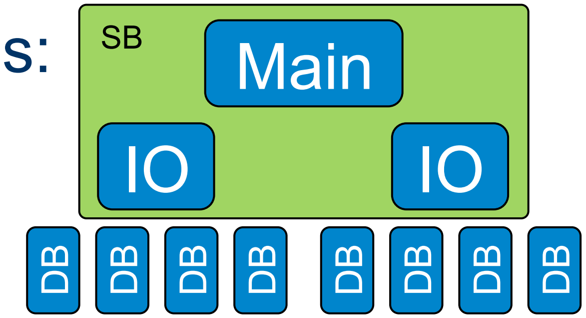
BERKELEY LAB

Singles Mode – DB Statemachine



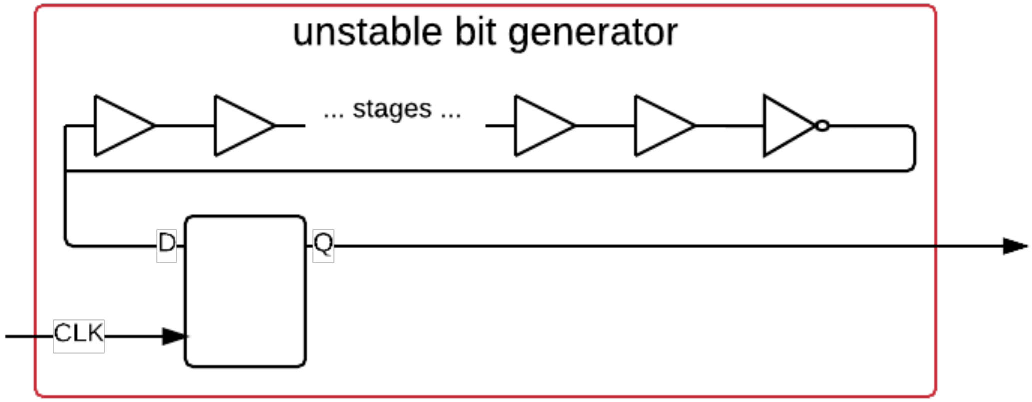
Singles Mode – Arbitrator Intro

- For a small system, i.e., single chassis:
- Arbitration occurs in SB at:
  - IO FPGAs: 4-in-1-out (loss of data)
  - Main FPGA: 2-in-1-out (loss of data)



- Reusable code on Main and IO FPGAs on SB.
- Arbitrator uses a true-random-number generator to select one of the candidates.
- No priority is given to any slot.
- Arbitrator works for all cases 1-in-1-out, 2-in-1-out, 3-in-1-out, and 4-in-1-out.
- Only 1 clock cycle required for selection.

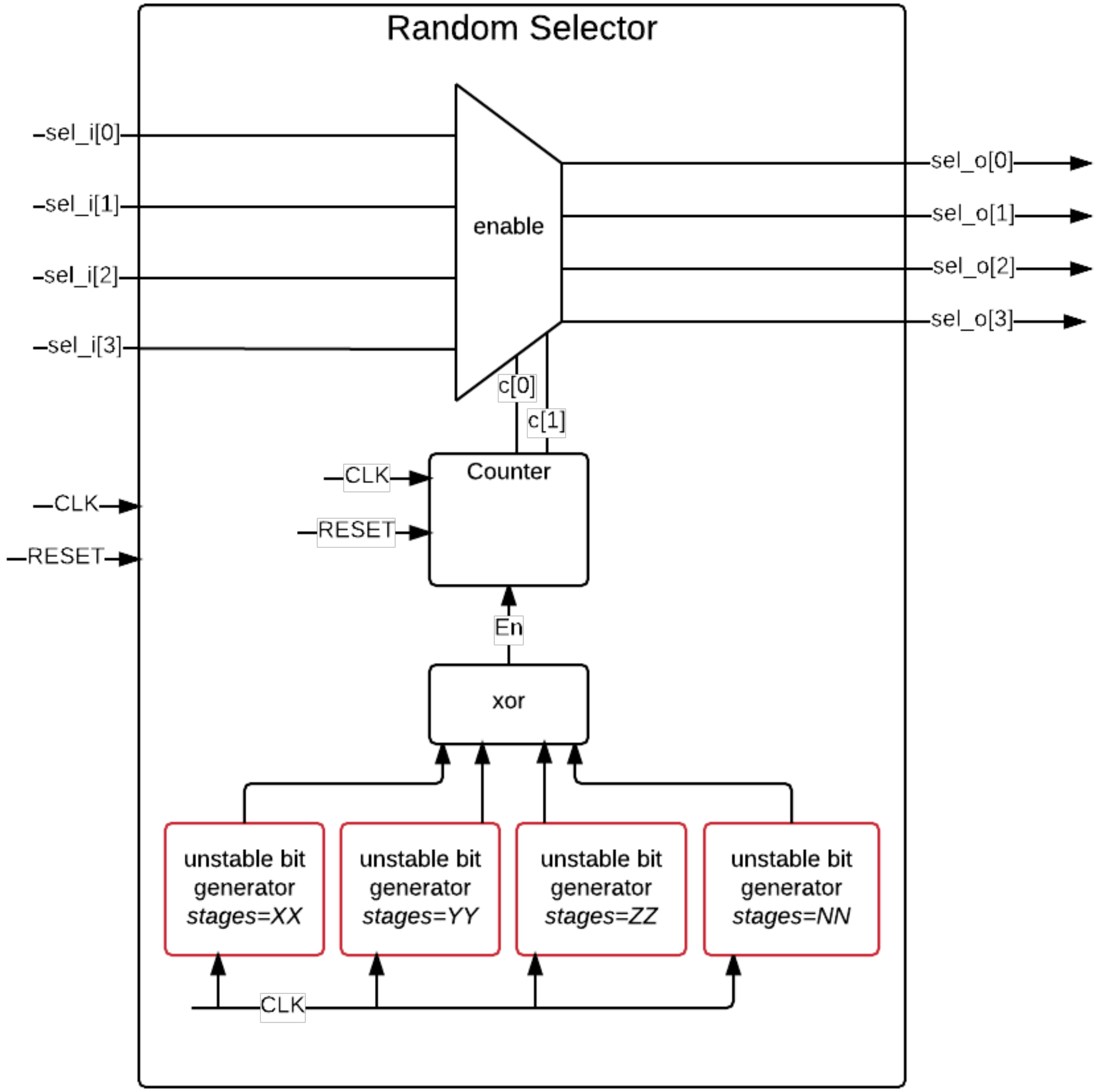
Singles Mode – Unstable Bit Generator



- Uses ring oscillator.
- Output bit (i.e. Q) is unstable by design.
- Randomness is created from voltage, temperature, and near-by-logic variations.
- Number of ring oscillator stages is programmable.

Singles Mode – TRNG and Arbitrator

- 4 instances of UBG
- Different stages for each instance.
- UBGs outputs are XOR'd to add more randomness.
- XOR output enables/disables a free running counter.
- Variable Selection Design:
- Counter maximum value the Least Common Multiple of the number of inputs (4 for IO and 2 for Main)
- Counter output is wrapped to create a fair selection. Wrapping is done using the modulo of the number of active candidates.



Singles Mode – Binary file format

- Headers are prepended to singles data.
  - Time of Acquisition.
  - All acquisition Settings.
- 4KB are reserved for user defined content.
- Simple 32-bit format for easy parsing and manipulation.

Offset	[31..24]	[23..16]	[15..8]	[7..0]	Comment
0	Epoch timestamp				Local Time
1	Reserved				
2	Acquisition Duration				From Software
3	Acquisition Mode Payload				See cmd id 0x0003
4	Acquisition Mode Settings Payload				See cmd id 0x0005
5	Reserved				future data format
6	Misc.Software Settings				processed/raw?
7	Reserved				
...					
9					
10	User Defined				
...					
999					
B=1000	Singles Packet (0)				Event can originate
B+1	Singles Packet (1)				from any DB.
	...				
B+N-1	Singles Packet (B+N-1)				N = Slice Width
B+N	Singles Packet (0)				Event can originate
B+N+1	Singles Packet (1)				from any DB.
	...				

Singles Mode – Example

- OpenPET first provided example. More coming!
- Very basic example.
- Computes the energy, i.e., area under the curve.
- Utilizes 5 pipeline stages; up to 16 data points.
- 128-bit Word packs:
  - 6-bit energies 16 channels = 6*16 bits.
  - 9-bit address.
  - 4-bit hit trigger counter.
  - 4-bit packet id for 4 packets = 4*4 bits.

Singles Mode – Energy Example Word

N=0, [31:0]	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N=1, [63:32]	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
N=2, [95:64]	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64
N=3, [127:96]	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96

Starting from least significant bit (LSB)

- (2:0) Detector Board Address (populated by parent)
- (5:3) DU Address (populated by parent)
- (8:6) MB Address (populated by parent)
- (12:9) Number of channels that triggered
- (15:13) Unused
- (21:16) Channel0 Energy
- (27:22) Channel1 Energy
- (31:28) Packet ID

Singles Mode – Energy Example Settings

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Starting from least significant bit (LSB)

- (3:0) Total number of ADC clock ticks to finish a single Event computation (2^4 -1 = 15)
- (7:4) Reserved, max adc clock ticks to process data = 2^8-1 = 255
- (15:8) Reserved
- (31:16) Not Used

Notes:

- Event computation clocks ticks should be greater than 1.
  - Number of pipeline stages is pre-defined in the firmware as a constant.
  - PipelineStages = ceil(EventCompuationClockTicks/SliceWidth) + 1

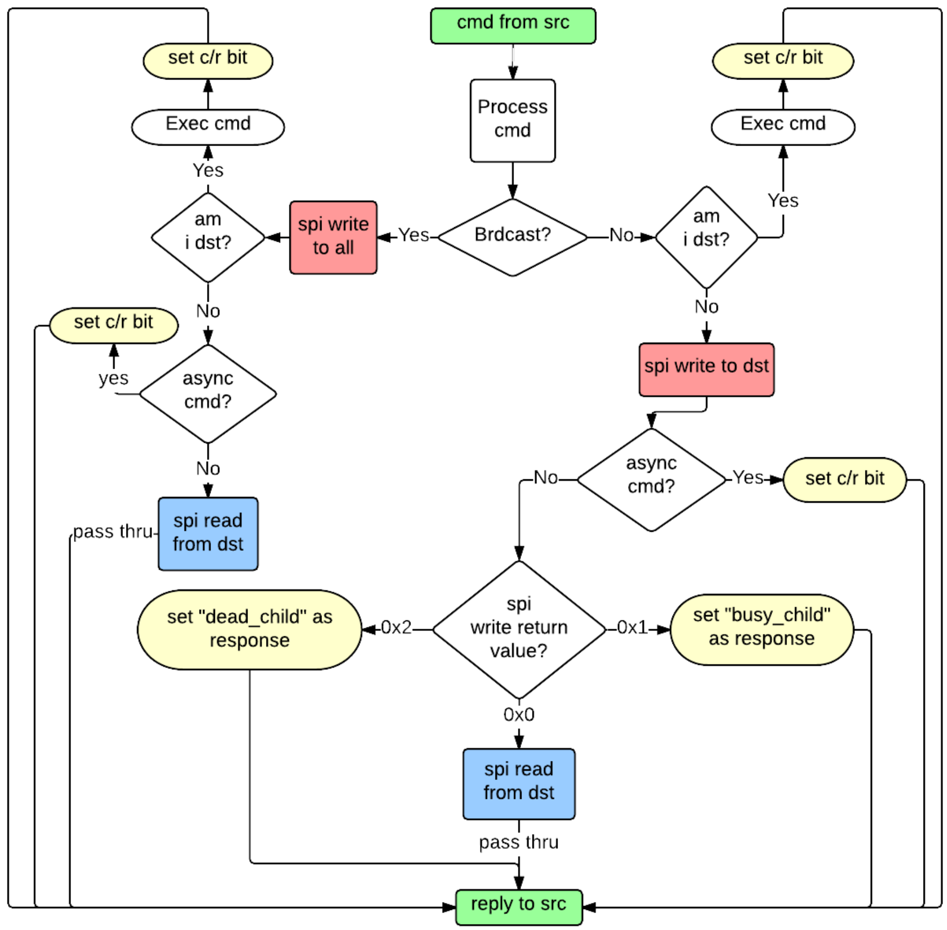
Firmware – Throughput

- Single DB to SB 40 MHz * 32-bits = 1280 Mbps.
- USB 2.0 max theoretical throughput is 480 Mbps.
  - QuickUSB max measured throughput is 330 Mbps.
- GbE max theoretical throughput is 1250 Mbps.
  - *Expected* throughput is 500 Mbps (no optimization).
- SB throttles DB(s) and IO(s) to adapt to output speed.

Embedded Software Changes I

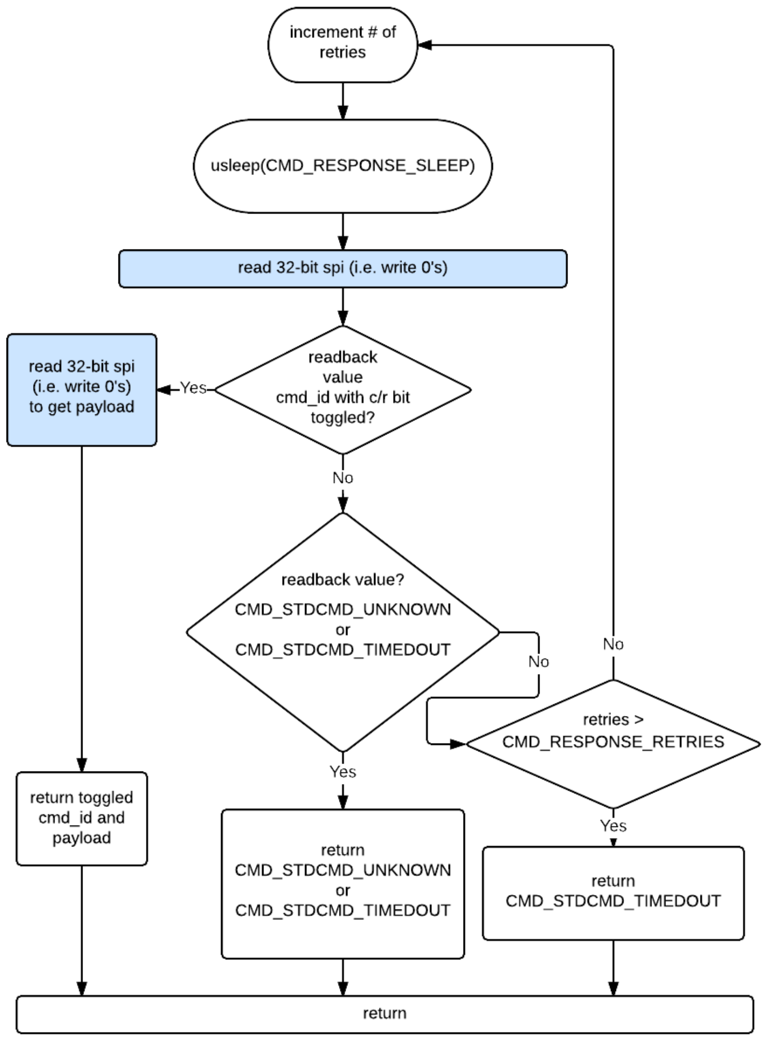
- Unified architecture on all FPGAs:
  - Minimalistic NIOS II Microprocessor on all FPGAs.
  - Standard 32-bit SPI interface between all nodes.
    - Can broadcast to all slaves.
  - Standard Software-Firmware interface on all FPGAs.
  - Interrupt based processing.
    - Interrupt Service Routine (ISR) is standard across all NIOS IIs.
- Automated builds; TCL scripts are used to generate BSP at compile time.
- Two types of commands; synchronous and asynchronous, i.e., non-blocking commands.
- No C++, just plain C. Reduces code based size by 70%.

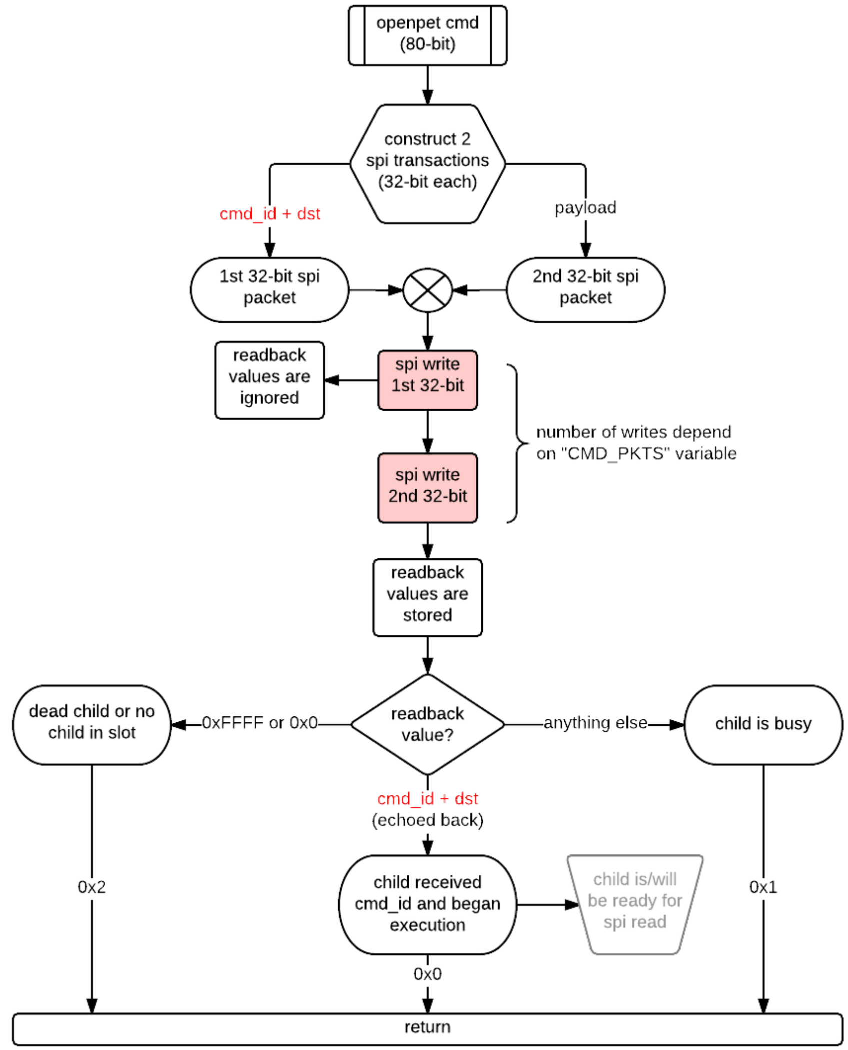
SPI Flow



BERKELEY LAB

SPI read() and write()





Embedded Software Changes II

- SupportBoard
  - Main FPGA
    - Child FPGA images are not compressed. Users don't need to modify SB Embedded Software.
    - Generic QuickUSB interface. Parameterization of OpenPET command length, time-to-live, timeout, etc.
    - Ethernet compatible code base.
    - UART-over-JTAG console for additional verbosity.
  - IO FPGAs
    - Embedded Software is embedded in firmware image (bitstream) at build time.
    - *Extremely* small NIOS-II. Code size is 7KB.

Embedded Software Changes III

- DetectorBoard
  - Embedded Software is embedded in firmware image (bitstream) at build time.
  - Extremely small NIOS-II. Code size is 11KB.
  - New peripheral device drivers for ADC, DAC, and SRAM.
  - Direct access to all ADC and DAC registers.

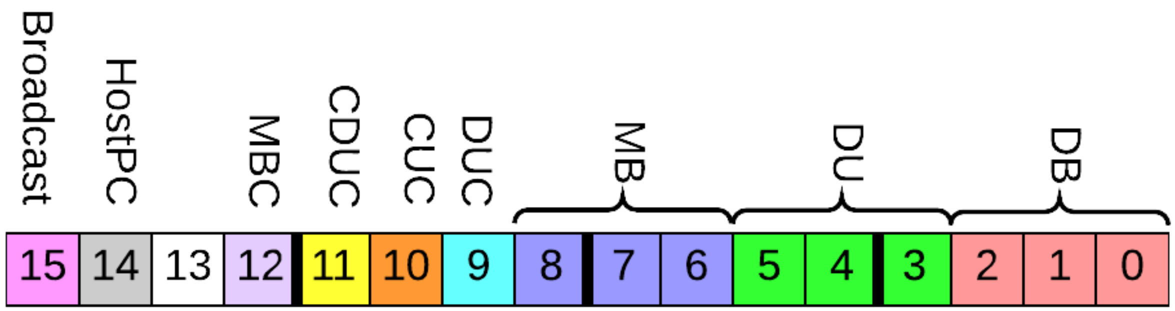
Software Changes

- All new command architecture. Old commands won't work.
- Generic user-defined command length (currently set at 80-bits like v1.0)
- New cross-platform scripting interface. Supports Microsoft Windows, Mac OS X, and GNU/Linux.
- New openpet executable for Windows x64.
- Many example cross-platform scripts for Scope and Singles modes.
- Optional real-time user defined data processing.
- Multiprocessing and queuing are used to achieve maximum throughput regardless of storage speed.

Commands

CMD ID (16 bits)	SRC Addr (16 bits)	DST Addr (16 bits)	Payload (32 bits)

- Generic 80-bits OpenPET commands are used. Can be increased to any value.
- 2^{14} possible commands IDs.
- 24 commands are currently available.
- Each command ID has unique 32-bit payload.
- Addressing for all nodes in the platform.



Command-line interface

```
Administrator: C:\Windows\system32\cmd.exe

C:\Users\FTAbunimeh\openpet\faisal\hostpc\sw>dist\openpet\openpet.exe
usage: openpet.exe [-h] [-v] [-l] [-d DEVICEINDEX]
                   [-c ID DST PAYLOAD | -a DURATION | -sr FILE DST SIZE OFFSET |
 -sw FILE DST OFFSET]
                   [-o FILE] [-t TIMEOUT] [-n RETRIES]

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Show debugging info.
  -l, --list            List quickusb devices.
  -d DEVICEINDEX, --device DEVICEINDEX
                        Quickusb device index. List devices to see indexes.
                        DEFAULT=0
  -c ID DST PAYLOAD, --command ID DST PAYLOAD
                        Sends a command to a destination module with a
                        specific payload.
  -a DURATION, --acquire DURATION
                        Acquire data for the specified duration (in seconds).
                        Partial seconds are OK.
  -sr FILE DST SIZE OFFSET, --sram-read FILE DST SIZE OFFSET
                        Reads SRAM contents from OpenPET and writes it to
                        FILE.
  -sw FILE DST OFFSET, --sram-write FILE DST OFFSET
                        Writes FILE contents to SRAM.
  -o FILE, --outputfile FILE
                        File name to save acquired data.
  -t TIMEOUT, --timeout TIMEOUT
                        Timeout duration (in seconds) between retries. Partial
                        seconds are OK. DEFAULT=0.200
  -n RETRIES, --retries RETRIES
                        Number of times I should try to contact OpenPET System
                        before giving up. DEFAULT=20
```

New *openpet.exe* executable for Windows x64.
Executable is basically openpet.py built with pyinstaller.

Simple command flow

- Set the acquisition mode to scope. Note that the MSB in DST is set to 1 to denote broadcast to all nodes:

```
$ openpet -c 3 0x8003 1
```

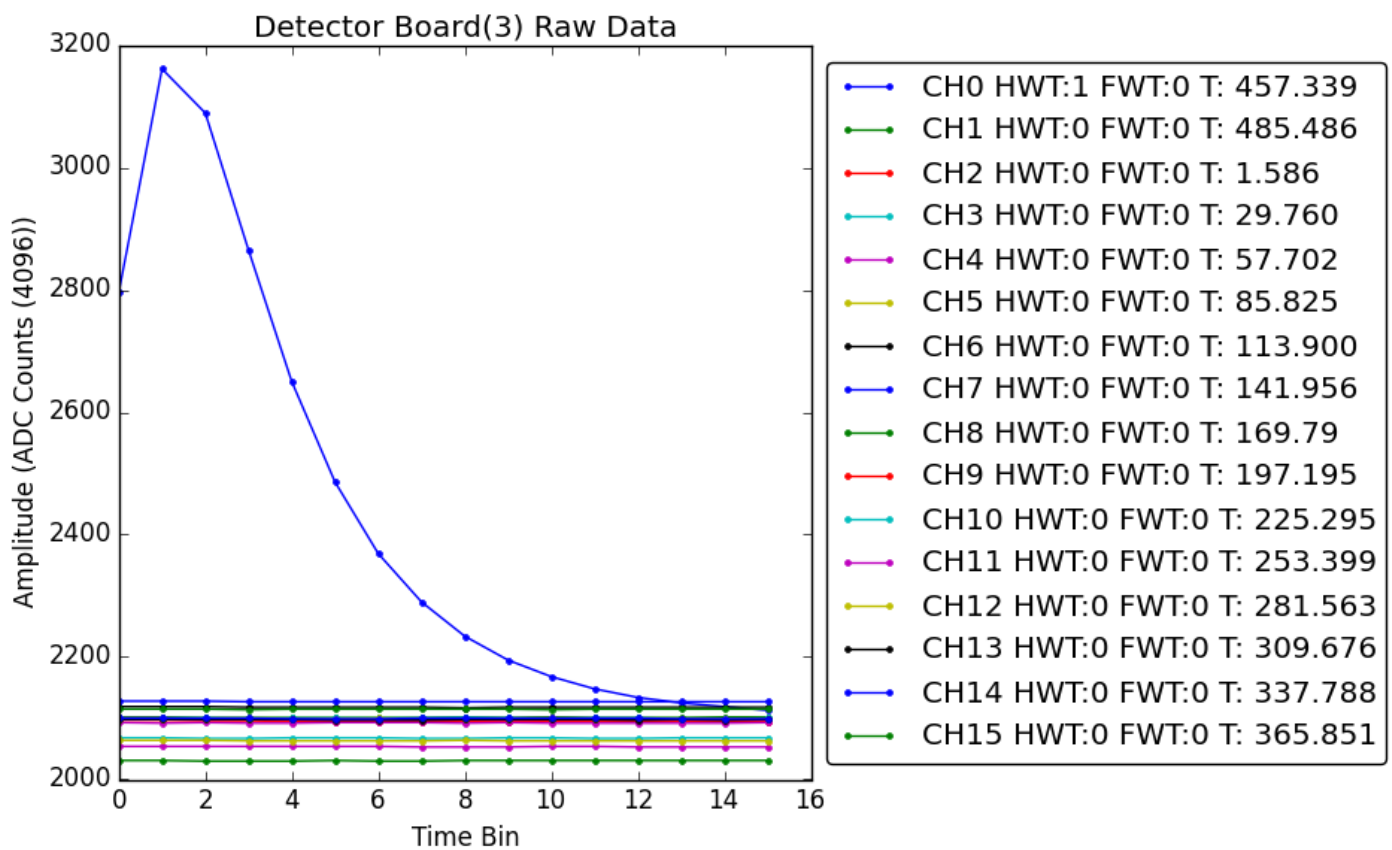
- Set scope settings to 0x02000101. Number of Samples is 16, samples before trigger is 0, and trigger window is 2:

```
$ openpet -c 5 0x8003 0x02000100
```

- Acquire scope data for 10 seconds, save acquired data to auto-generated unique file name.

```
$ openpet -a 10
```

Scope Output



Scripting Interface

- Python library provided.
- Can embedded in many languages:
  - https://docs.python.org/2/extending/embedding.html
- Interface provides low level access to all return codes, errors, raw data, etc.
- Cross platform.
- Uses numpy for real-time analysis, computing, and handling "Big Data."

Untitled

File Edit Format Run Options Window Help

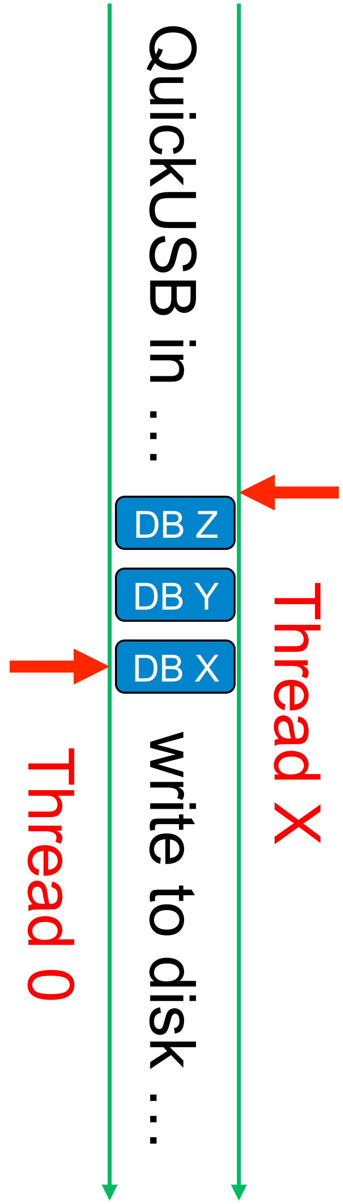
```python
#!/usr/bin/env python
from OpenPETlib import *

def main():
    op = OpenPET() # create OpenPET object
    # we will use the default QuickUSB deviceIndex i.e. 0
    # If you want to use deviceIndex 2 call use the follow:
    # op.qusb = op.init_qusb(2)
    op.qusb = op.init_qusb()

    # Set some parameters here, or anywhere in the script
    op.d_acq_t = 10 # data acquisition duration (seconds), partial seconds are OK.

    # NOTE: ADC and DAC are configured on powerup by default.
    # See other example scripts for more configuration options.

    # Target Address
    # Broadcast bit is sit to '1' to allow all nodes to see the commands we are sending out
    # The reply to these commands will come from DB located @ 0x0003
    trgt = 0x8003

    # Set Mode to Scope
    op.openpet_cmd(0x0003, trgt, 0x00000001)

    # Set Scope mode settings
    # 0x02000100 = 16 samples, 0 before trigger, and 2 trigger window,
    oscpayload = 0x02000101
    op.openpet_cmd(0x0005, trgt, oscpayload)
    cmd, srcp, dst, pyld = op.openpet_cmd(0x0006, trgt, 0) # Read them back
    logging.debug('returned PAYLOAD 0x%08X ', pyld)
    if (pyld == oscpayload):
        logging.debug('Correct! Scope mode settings read back')
    else:
        logging.error('Wrong! Scope mode settings read back')
        sys.exit()

    # Start Acquisition
    # getdata() will set the Acquisition Action to 'Run' by default
    # if you want to manually set it, see other examples
    logging.info('Acquiring data for %d seconds... ', op.d_acq_t)
    op.getdata()
    # getdata() will set the Acquisition Action to 'Reset' when done by default
```

Ln: 1 Col: 0

BERKELEY LAB

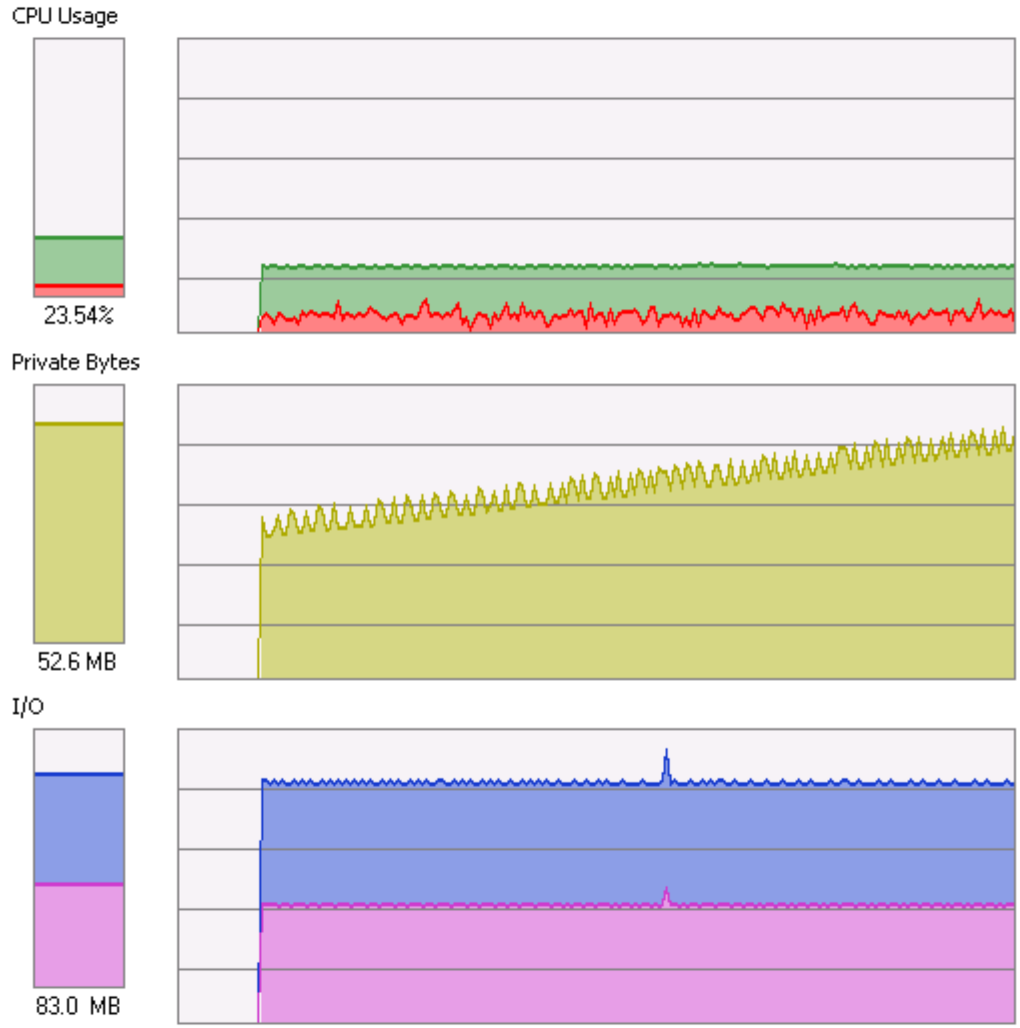
Multiprocessing and Queuing

- Utilizes Python multiprocessing queuing.
- Uses Maximum, CPUs, Cores, and Threads regardless of what hardware architecture you have.
- Incoming QuickUSB (Ethernet in the future) data is queued up into RAM.
- Each data block is handled by an independent thread.
- Data is written to storage (local or remote), thread is destroyed, and memory is freed.



Handling High Speed Data – Fast

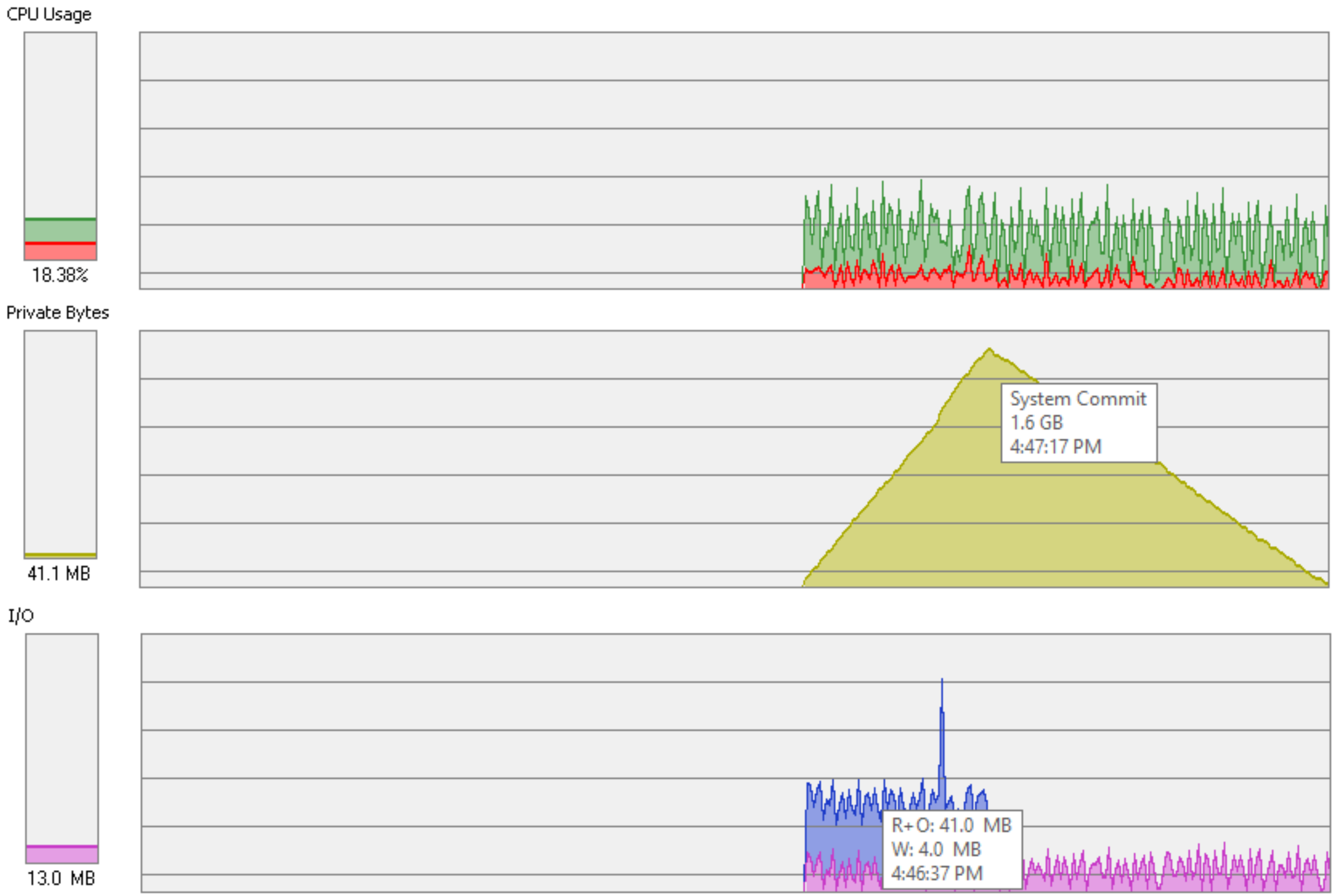
- Test setup:
  - OpenPET v2.0
  - 20 MHz trigger rate.
  - Acquisition time 200s
  - 16 samples
- Data rate in = 42 MB/s (quickusb)
- Data rate out = 42 MB/s (disk)



Handling High Speed Data – Slow I

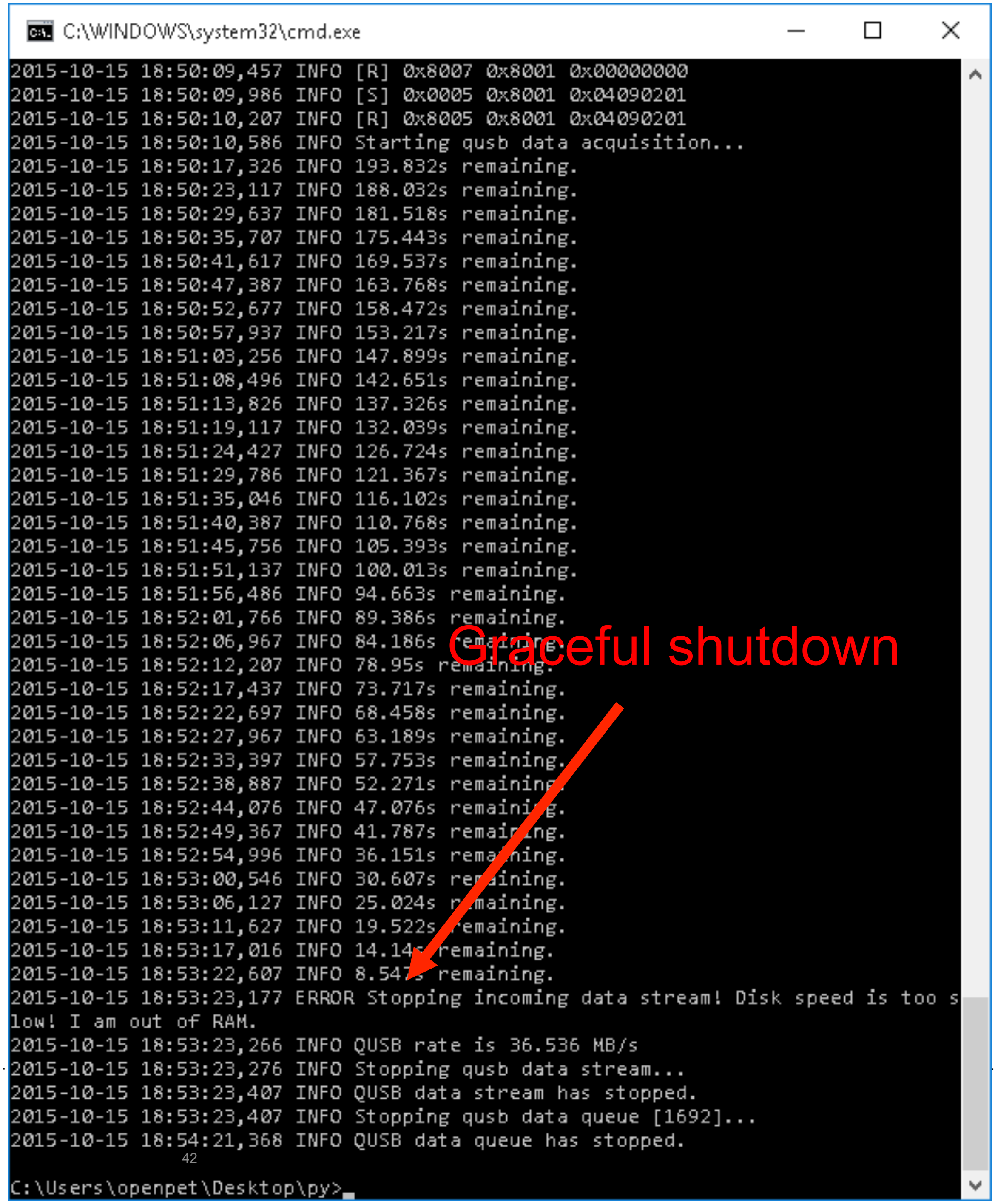
- Test setup:
  - OpenPET v2.0
  - 20 MHz trigger rate.
  - Acquisition time 200s
  - 16 samples
- Data rate in = 42 MB/s (quickusb)
- Data rate out = 30 MB/s (disk)

Handling High Speed Data – Slow II



Handling High Speed Data – Slow III

- Recovers all collected data.
- Recorded file maintains integrity.
- User is notified of error.



Release Engineering Plans

- RERO: Release early, release often.
- Open and transparent development. Work closer with OpenPET community.
- Faster life cycle of firmware, embedded software, and software. Agile and Lean!
- New features will be added to major revisions i.e. v3.0, v4.0, etc.
  - Probable breakage of ABI (Application Binary Interface) and API (Application Programming Interface)
  - Code freeze 1 month before release for testing.
- Minor revisions v2.1, v2.2, etc. will be mainly bug fixes, small features.
  - Stable. No breakage of ABI. Probable changes to API.

BERKELEY LAB

Thanks!

Questions?